# CH583/CH582 Datasheet

V1.9

## Overview

The CH583 is a RISC-V MCU (SoC) integrated with BLE wireless communication. It integrates 2Mbps Bluetooth Low Energy communication module, 2 full-speed USB host and device controllers and transceiver, 2 SPIs, 4 UARTs, 14-channel ADC, touch-key detection module, RTC and other peripheral resources.

## Features

- **Royalty-Free core:**
- 32-bit QingKe RISC-V processor WCH RISC-V4A
- Support RV32IMAC instruction set, hardware multiplication and division
- Low-power 3-stage assembly line, high-performance 1.56DMIPS
- Multi-speed system frequency: minimum 32 KHz
- Unique high-speed interrupt response mechanism
- **512K-byte non-volatile memory FlashROM:**
- 448KB user application program memory area CodeFlash
- 32KB user non-volatile data memory area DataFlash
- 24KB system boot program memory area BootLoader
- 8KB system non-volatile configuration information memory area InfoFlash
- Support ICP, ISP and IAP, support OTA wireless upgrade
- Basically, zero-wait at the system frequency of 20MHz
- **32K-byte volatile data storage SRAM:**
- Sleep retention memory area RAM30K using 30KB dual power supply
- Sleep retention memory area RAM2K using 2KB dual power supply
- **Power management and low power:**
- Support 3.3V and 2.5V power, CH583M supports 1.8V
- Built-in DC-DC conversion, reducing power consumption
- Idle mode: 1.6mA
- Halt mode: 320uA
- Sleep mode: Multiple gears from 0.7uA to 2.8uA
- Shutdown mode: Multiple gears from 0.2uA to 2.3uA
- Optional low-voltage monitoring of low-power or high-precision battery voltage
- **Security properties:**
- AES-128 encryption and decryption, unique chip ID
- **Bluetooth Low Energy (BLE):**
- Integrated with 2.4GHz RF transceiver and baseband and link control
- -98dBm RX sensitivity, programmable +6dBm TX power
- BLE complies with Bluetooth Low Energy Specification 5.0
- Support 2Mbps, 1Mbps, 500Kbps and 125Kbps
- The wireless communication distance is about 500 meters at 6dBm TX power
- The communication distance is about 1000 meters at the TX power of 6dBm and 125Kbps
- Provide optimized protocol stack and application layer API, and support networking
- **Real-time clock (RTC):**
- 2 modes of timing and triggering
- **Universal serial bus (USB):**
- 2 sets of independent USB controller and transceiver
- 15 endpoints, support DMA, support 64-byte data packet
- Integrated USB 2.0 full-speed transceiver PHY, no peripheral required
- Support host and device mode at full-speed/low-

speed

- **Analog to digital converter (ADC):**
- 12-bit analog to digital converter, support differential and single-ended input
- 14 external analog signal channels and 2 internal signal channels
- **Touchkey detection module (TouchKey):**
- 14-channel
- **Timer and pulse width modulation (PWM):**
- 4 sets of 26-bit timers, which can reach 4.2S at the frequency of 16MHz
- 4-channel capture/sample, support rising edge/falling edge /double edge
- 4-channel 26-bit PWM output, 8-channel 8-bit PWM output
- **Universal asynchronous receiver/transmitter (UART):**
- 4 independent UARTs, compatible with 16C550, built-in 8-level FIFO
- 23-bit counter, up to 6Mbps communication baud rate
- UART0 supports some modems, supports hardware automatic flow control
- UART0 supports automatic matching of slave address during multi-device communication
- **Serial peripheral interface (SPI):**
- 2 sets of independent SPIs, built-in FIFO
- SCK serial clock frequency can reach half of system clock frequency
- SPI0 supports Master and Slave modes, DMA capability
- **2-wire serial interface (I2C):**
- Support Master and Slave modes, compatible with SMBus
- Support 7-bit or 10-bit address and bus broadcast
- Support arbitration, error detection, PEC check and clock extension
- **Temperature sensor (TS)**
- **Clock:** Built-in PLL, built-in 32KHz clock
- **General purpose input/output (GPIO):**
- 40 GPIOs, 2 of them support 5V signal input
- Optional pull-up or pull-down resistor, optional output drive capability
- 32 GPIOs support level or edge interrupt input
- 32 GPIOs support level or edge wake-up input
- **Package:** QFN48_5X5, QFN28_4X4

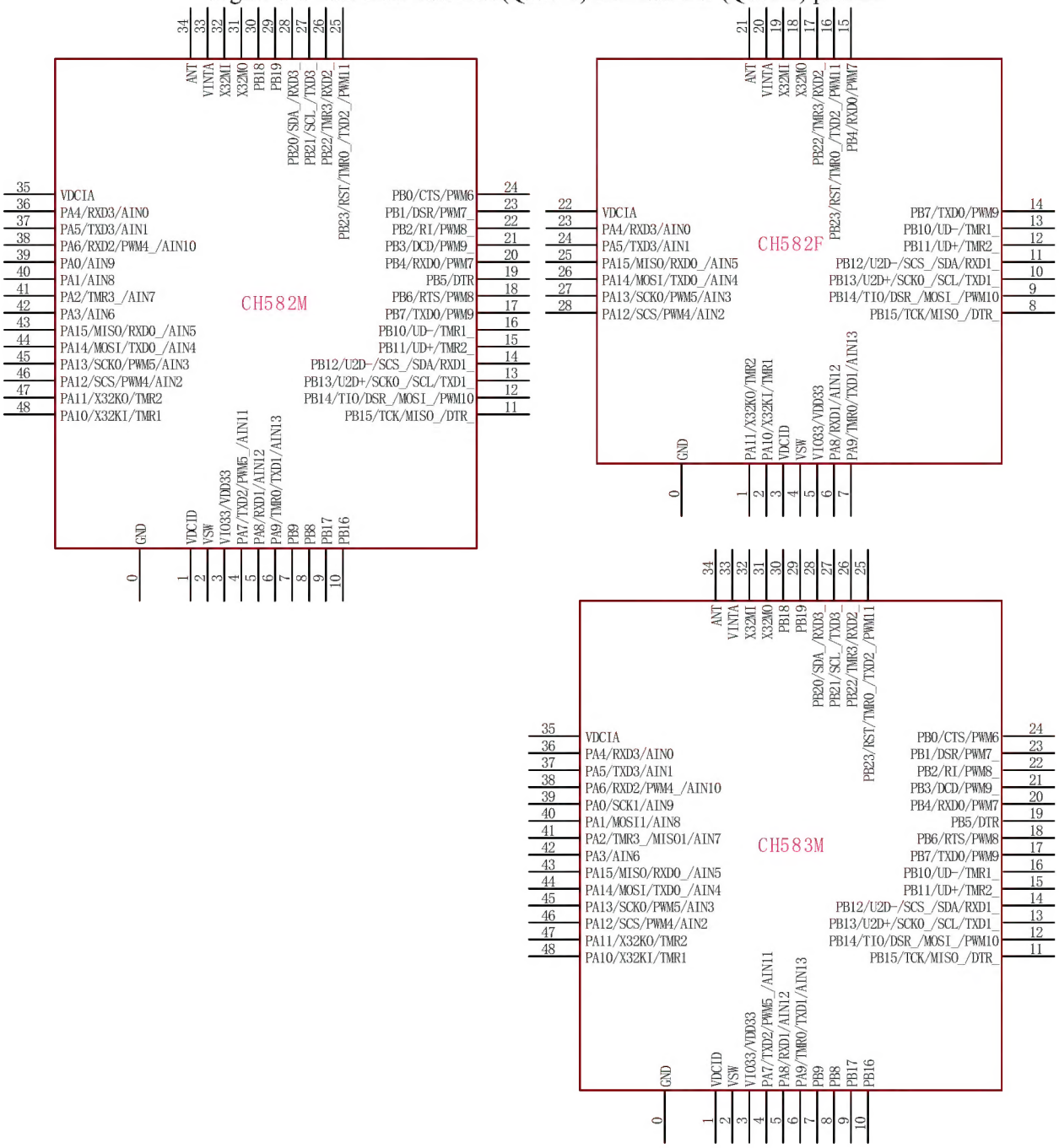
Compared with CH582, CH583 adds SPI1 master and supports at least 75V power voltage.

| Part No. | CodeFlash +BootLoader +DataFlash | RAM | RTC | Timer | Capture | PWM | UART | SPI | I2C master slave | BLE | ADC and TS | Capacitive Touch-key | USB host | USB device | DC - DC | Min. supply voltage | GPIO | Package |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH583M | 448+24+32K | 30+2K | √ | 4 | 4 | 4+8 | 4 | 2 | √ | √ | 14+1 | 14 | 2 | 2 | √ | 1.75 | 40 | QFN48 |
| CH582M | 448+24+32K | 30+2K | | | 4 | 4+8 | | master | √ | | 14+1 | 14 | 2 | 2 | √ | 2.3 | 40 | QFN48 |
| CH582F | 448+24+32K | 30+2K | | | 4 | 4+6 | | -slave | √ | | 8+1 | 8 | 2 | 2 | √ | 2.3 | 20 | QFN28 |

# Chapter 1 Pinouts and Pin Definitions

## 1.1 Pinouts

Figure 1-1 CH582M/CH583M(QFN48) and CH582F (QFN28) pinouts



## 1.2 Pin definitions

| QFN48 Pin No. | QFN28 Pin No. | Pin Name | Pin Type | Alternate function takes precedence | Description |
|---|---|---|---|---|---|
| 0 | 0 | GND | Power | - | EPAD, ground, voltage 0V as reference point |
| 1 | 3 | VDCID | Power | - | Power input of internal digital circuit LDO regulator, requires an external decoupling capacitor. 2.2uF is recommended (0.47uF~4.7uF supported, small capacitance causes little power saved and BLE sensitivity reduced 2dBm) when DC-DC is enabled. While 0.1uF or more is recommended when DC-DC is disabled. |
| 2 | 4 | VSW | Power | - | Internal DC-DC switch output, must be connected to VDCID with inductor in series close to the pin when DC-DC is enabled, and 10uH inductor is recommended (3.3uH~33uH supported). While it can be directly connected to VDCID when DC-DC is disabled. |
| 3 | 5 | VDD33 | Power | VBAT | DC-DC or battery power input, requires an external decoupling capacitor close to pin. 2.2uF or 1uF is recommended when DC-DC is enabled, while 0.1uF is recommended when DC-DC is disabled. |
| | | VIO33 | Power | - | I/O power input, need to be close to the pin external decoupling capacitor, can share the same capacitor with VDD33. |
| 4 | None | PA7 | I/O/A | TXD2<br>/PWM5_<br>/AIN11 | PA7: General-purpose bidirectional digital I/0 pin.<br>TXD2: UART2 serial data output.<br>PWM5_: PWM output channel5 pin mapping.<br>AIN11: ADC analog signal input channel 11. |
| 5 | 6 | PA8 | I/O/A | RXD1<br>/AIN12 | PA8: General-purpose bidirectional digital I/0 pin.<br>RXD1: UART1 serial data input.<br>AIN12: ADC analog signal input channel 12. |
| 6 | 7 | PA9 | I/O/A | TMR0<br>/TXD1<br>/AIN13 | PA9: General-purpose bidirectional digital I/0 pin.<br>TMR0: Capture input0 and PWM output channel0 of Timer0.<br>TXD1: UART1 serial data output.<br>AIN13: ADC analog signal input channel 13. |
| 7 | None | PB9 | I/O | - | PB9: General-purpose bidirectional digital I/0 pin. |
| 8 | None | PB8 | I/O | - | PB8: General-purpose bidirectional digital I/0 pin. |
| 9 | None | PB17 | I/O | - | PB17: General-purpose bidirectional digital I/0 pin. |
| 10 | None | PB16 | I/O | - | PB16: General-purpose bidirectional digital I/0 pin. |
| 11 | 8 | PB15 | I/O<br>/5VT | TCK<br>/MISO_<br>/DTR_ | PB15: General-purpose bidirectional digital I/0 pin<br>TCK: Serial clock input of 2-wire simulation debug interface. Note3<br>MISO_: MISO pin mapping of SPI0.<br>DTR_: DTR pin mapping of UART0. |

| | | | | | |
|---|---|---|---|---|---|
| 12 | 9 | PB14 | I/O<br>/5VT | TIO<br>/MOSI_<br>/PWM10<br>/DSR_ | PB14: General-purpose bidirectional digital I/0 pin.<br>TIO: Serial data input/output of simulation debug interface, with built-in pull-up. Note 3<br>MOSI_: MOSI pin mapping of SPI0.<br>PWM10: PWM output channel10. DSR_: DSR pin mapping of UART0. |
| 13 | 10 | PB13 | I/O | U2D+<br>/SCK0_<br>/SCL<br>/TXD1_ | PB13: General-purpose bidirectional digital I/0 pin.<br>U2D+: D+ data line of USB2 bus.<br>SCK0_: SCK pin mapping of SPI0.<br>SCL: I2C serial clock pin, master output and input/slave input.<br>TXD1_: TXD1 pin mapping of UART1. |
| 14 | 11 | PB12 | I/O | U2D-<br>/SCS_<br>/SDA<br>/RXD1_ | PB12: General-purpose bidirectional digital I/0 pin.<br>U2D-: D- data line ofUSB2 bus.<br>SCS_: SCS pin mapping of SPI0.<br>SDA: I2C serial data pin, open-drain output and input.<br>RXD1_: RXD1 pin mapping of UART1. |
| 15 | 12 | PB11 | I/O/A | UD+<br>/TMR2_ | PB11: General-purpose bidirectional digital I/0 pin.<br>UD+: D+ data line of USB bus.<br>TMR2_: TMR2 pin mapping of Timer2. |
| 16 | 13 | PB10 | I/O/A | UD-<br>/TMR1_ | PB10: General-purpose bidirectional digital I/0 pin.<br>UD-: D- data line of USB bus.<br>TMR1_: TMR1 pin mapping of Timer1. |
| 17 | 14 | PB7 | I/O | TXD0<br>/PWM9 | PB7: General-purpose bidirectional digital I/0 pin.<br>TXD0: UART0 serial data output.<br>PWM9: PWM output channel9. |
| 18 | None | PB6 | I/O | RTS<br>/PWM8 | PB6: General-purpose bidirectional digital I/0 pin.<br>RTS: UART0 MODEM output signal, request to send.<br>PWM8: PWM output channel8. |
| 19 | None | PB5 | I/O | DTR | PB5: General-purpose bidirectional digital I/0 pin.<br>DTR： UART0 MODEM output signal, data terminal ready. |
| 20 | 15 | PB4 | I/O | RXD0<br>/PWM7 | PB4: General-purpose bidirectional digital I/0 pin.<br>RXD0: UART0 serial data input.<br>PWM7: PWM output channel7. |
| 21 | None | PB3 | I/O | DCD<br>/PWM9_ | PB3: General-purpose bidirectional digital I/0 pin.<br>DCD: UART0 MODEM input signal, carrier detect.<br>PWM9_: PWM output channel9 pin mapping. |
| 22 | None | PB2 | I/O | RI<br>/PWM8_ | PB2: General-purpose bidirectional digital I/0 pin.<br>RI: UART0 MODEM input signal, ring indicator.<br>PWM8_: PWM output channel pin mapping. |
| 23 | None | PB1 | I/O | DSR<br>/PWM7_ | PB1: General-purpose bidirectional digital I/0 pin.<br>DSR: UART0 MODEM input signal, data device ready.<br>PWM7_: PWM output channel7 pin mapping. |
| 24 | None | PB0 | I/O | CTS<br>/PWM6 | PB0: General-purpose bidirectional digital I/0 pin.<br>CTS: UART0 MODEM input signal, clear to send. |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | PWM6: PWM output channel6. |
| 25 | 16 | PB23 | I/O | RST#<br>/TMR0_<br>/TXD2_<br>/PWM11 | PB23: General-purpose bidirectional digital I/0 pin.<br>RST#: External reset input, active low, with built-in pull-up resistor.<br>TMR0_: TMR0 pin mapping of Timer0.<br>TXD2_: TXD2 pin mapping of UART2.<br>PWM11: PWM output channel11. |
| 26 | 17 | PB22 | I/O | TMR3<br>/RXD2_ | PB22: General-purpose bidirectional digital I/0 pin.<br>TMR3: Capture input3 of Timer3 and PWM output channel3.<br>RXD2_: RXD2 pin mapping of UART2. |
| 27 | None | PB21 | I/O | SCL_<br>/TXD3_ | PB21: General-purpose bidirectional digital I/0 pin.<br>SCL_: I2C serial clock pin mapping.<br>TXD3_: TXD3 pin mapping of UART3. |
| 28 | None | PB20 | I/O | SDA_<br>/RXD3_ | PB20: General-purpose bidirectional digital I/0 pin.<br>SDA_: I2C serial data pin mapping.<br>RXD3_: RXD3 pin mapping of UART3. |
| 29 | None | PB19 | I/O | - | PB19: General-purpose bidirectional digital I/0 pin. |
| 30 | None | PB18 | I/O | - | PB18: General-purpose bidirectional digital I/0 pin. |
| 31 | 18 | X32MO | I/A | - | Inverted output of high frequency oscillator HSE, connected to one end of external 32MHz crystal. |
| 32 | 19 | X32MI | A | - | Input of high frequency oscillator HSE, connected to the other end of external 32MHz crystal. |
| 33 | 20 | VINTA | Power | - | The power node of the internal analog circuit requires an external decoupling capacitor close to the pin. DC-DC is not enabled when it is recommended that 0.47uF or greater; DC-DC is enabled when 1uF is available to support 0.47uF ~ 2.2uF, the capacitance is large, the signal quality is good but the power consumption is slightly larger. |
| 34 | 21 | ANT | A | - | RF signal input and output, it is recommended to connect the antenna directly. |
| 35 | 22 | VDCIA | Power | - | Power input of internal analog circuit LDO regulator, requires an external decoupling capacitor. It is recommended not to be smaller than 0.1uF, and it is recommended to be connected to VDCID directly. |
| 36 | 23 | PA4 | I/O/A | RXD3<br>/AIN0 | PA4: General-purpose bidirectional digital I/0 pin.<br>RXD3: UART3 serial data input.<br>AIN0: ADC analog signal input channel 0. |
| 37 | 24 | PA5 | I/O/A | TXD3<br>/AIN1 | PA5: General-purpose bidirectional digital I/0 pin.<br>TXD3: UART3 serial data output.<br>AIN1: ADC analog signal input channel 1. |
| 38 | None | PA6 | I/O/A | RXD2<br>/PWM4_<br>/AIN10 | PA6: General-purpose bidirectional digital I/0 pin.<br>RXD2: UART2 serial data input.<br>PWM4_: PWM output channel4 pin mapping.<br>AIN10: ADC analog signal input channel 10. |

| | | | | | |
|---|---|---|---|---|---|
| 39 | None | PA0 | I/O/A | SCK1<br>/AIN9 | PA0: General-purpose bidirectional digital I/0 pin.<br>SCK1: SPI1 SPI0 serial clock pin, host output.<br>AIN9: ADC analog signal input channel 9. |
| 40 | None | PA1 | I/O/A | MOSI1<br>/AIN8 | PA0: General-purpose bidirectional digital I/0 pin.<br>MOSI1: SPI1 serial data pin, host output.<br>AIN8: ADC analog signal input channel 8 |
| 41 | None | PA2 | I/O/A | TMR3_<br>/MISO1<br>/AIN7 | PA0: General-purpose bidirectional digital I/0 pin.<br>TMR3_: TMR3 pin mapping of Timer3.<br>MISO1: SPI1 serial data pin, host input.<br>AIN7：ADC analog signal input channel 7. |
| 42 | None | PA3 | I/O/A | AIN6 | PA3: General-purpose bidirectional digital I/0 pin.<br>AIN6：ADC analog signal input channel 6. |
| 43 | 25 | PA15 | I/O/A | MISO<br>/RXD0_<br>/AIN5 | PA15: General-purpose bidirectional digital I/0 pin.<br>MISO: SPI0 serial data pin, host input/slave output.<br>RXD0_: RXD0 pin mapping of UART0.<br>AIN5: ADC analog signal input channel 5. |
| 44 | 26 | PA14 | I/O/A | MOSI<br>/TXD0_<br>/AIN4 | PA14: General-purpose bidirectional digital I/0 pin.<br>MOSI: SPI0 serial data pin, host output/slave input.<br>TXD0_: TXD0 pin mapping of UART0.<br>AIN4: ADC analog signal input channel 4. |
| 45 | 27 | PA13 | I/O/A | SCK0<br>/PWM5<br>/AIN3 | PA13: General-purpose bidirectional digital I/0 pin.<br>SCK0: SPI0 serial clock pin, host output/slave input.<br>PWM5: PWM output channel 5.<br>AIN3: ADC analog signal input channel 3. |
| 46 | 28 | PA12 | I/O/A | SCS<br>/PWM4<br>/AIN2 | PA12: General-purpose bidirectional digital I/0 pin.<br>SCS: SPI0 chip select input in slave mode, active low.<br>PWM4: PWM output channel 4.<br>AIN2: ADC analog signal input channel 2. |
| 47 | 1 | PA11 | I/O/A | X32KO<br>/TMR2 | PA11: General-purpose bidirectional digital I/0 pin.<br>X32KO: Inverted output of low frequency oscillator, connected to the end of external 32KHz crystal.<br>TMR2: Capture input 2 and PWM output channel 2 of timer 2 |
| 48 | 2 | PA10 | I/O/A | X32KI<br>/TMR1 | PA10: General-purpose bidirectional digital I/0 pin.<br>X32KI: Input of low frequency oscillator, connected to the other end of external 32KHz crystal.<br>TMR1: Capture input 1 and PWM output channel 1 of timer 1. |

**Note:**

*(1) Pin Type:*

*I=TTL/CMOS level Schmitt input; O=CMOS level 3-state output;*

*A= Analog signal input or output; 5VT= Support 5V signal voltage input.*

*(2) The alternate functions and mappings of pins are arranged in the table based on their priorities from high to low, and the GPIO function is with the lowest priority.*

*(3) 2-wire simulation debug interface is configured by ISP tool. After the simulation debug interface is enabled, PB15 and PB14 are only used as TCK and TIO, and will not be used as GPIO or peripheral alternate function pins. After the simulation debug interface is disabled, PB15 and PB14 can be used as GPIO and peripheral alternate function pins.*

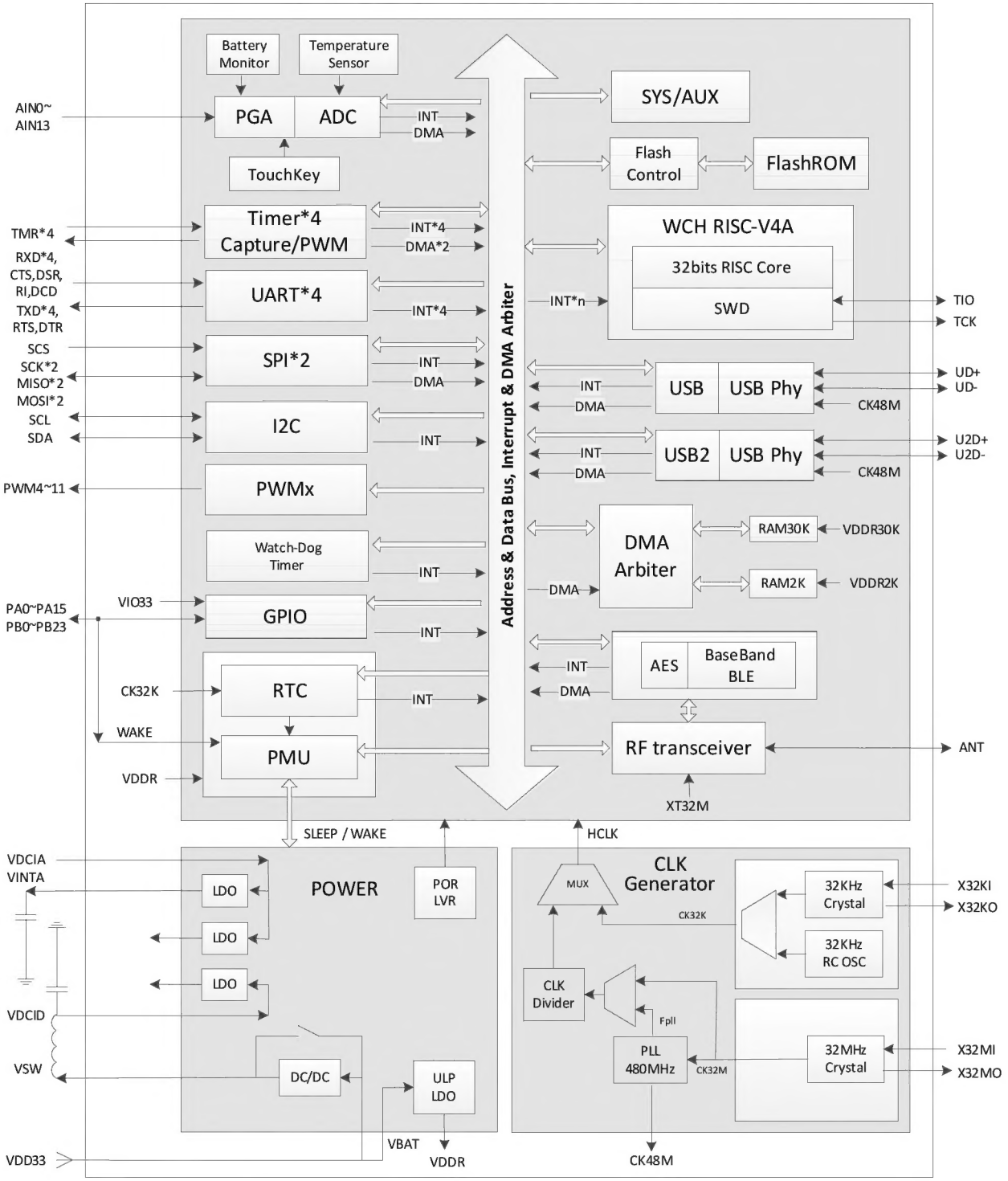
*(4) For the pins of those in other packages, please refer to the description of the pins with the same name in the table.*

# Chapter 2 System Architecture and Memory

## 2.1 System architecture

The following figure shows the system architecture block diagram of CH583. Its core is QingKe RISC-V microprocessor, please refer to QingKeV4 Processor Manual for details.

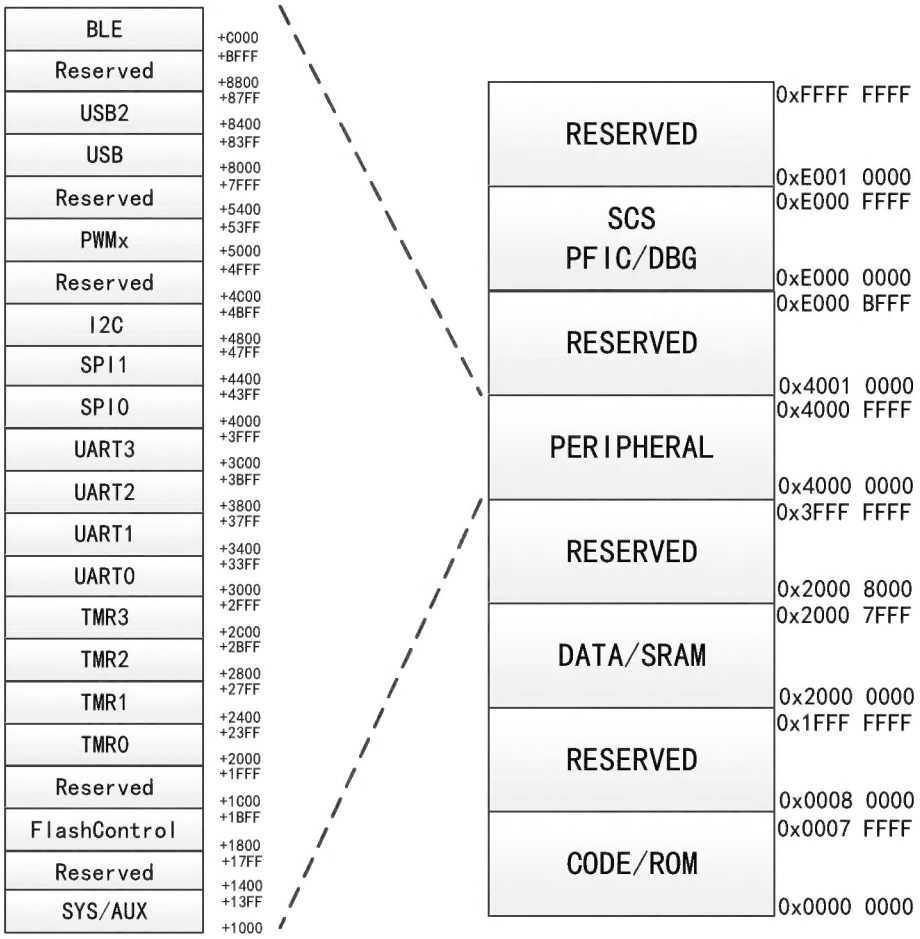
Figure 2-1 CH583 internal structure block diagram



## 2.2 Memory mapping

The addressing space of CH583 mainly includes several different areas, CODE area/FlashROM, DATA area/SRAM and peripherals, as shown in the figure below.

Figure 2-2 Memory mapping diagram




## 2.3 Memory mapping table

The address range of each memory mapping area is shown in the table below:

Table 2-1 Memory mapping area address

| Address range | Application | Description |
|---|---|---|
| 0x00000000-0x0007FFFF | On-chip CODE area, non-volatile memory | 512KB，FlashROM |
| 0x00080000-0x1FFFFFFF | Reserved | - |
| 0x20000000-0x20007FFF | On-chip DATA area, volatile memory | 32KB，SRAM |
| 0x20008000-0x3FFFFFFF | Reserved | - |
| 0x40000000-0x4000FFFF | Various peripherals | Multiple peripheral modules |
| 0x40010000-0xE000BFFF | Reserved | - |
| 0xE000C000-0xE000FFFF | Various peripherals in system | System Control Space (SCS) |
| 0xE0010000-0xFFFFFFFF | Reserved | - |

### 2.3.1 On-chip CODE area mapping table

Table 2-2 CODE area address

| Address range | Application | Description |
|---|---|---|
| 0x00000000-0x0006FFFF | User application program memory, CodeFlash | 448KB |
| 0x00070000-0x00077FFF | User non-volatile data memory, DataFlash | 32KB |
| 0x00078000-0x0007DFFF | System bootloader memory, BootLoader | 24KB |
| 0x0007E000-0x0007FFFF | System non-volatile configuration information memory, InfoFlash | 8KB |

The configuration information of the addresses 0x0007E000-0x0007EEFF can be set by the user through tools.

Table 2-3 Description of user non-volatile configuration information

| Bit address | Name | Application | Default Value |
|---|---|---|---|
| Bit2～bit0 | RESERVED | Reserved | 101b |
| Bit3 | CFG_RESET_EN | RST# external manual reset input pin enable | 0 |
| Bit4 | CFG_DEBUG_EN | Two-wire simulation debug interface SWD enable | 1 |
| Bit5 | RESERVED | Reserved | 0 |
| Bit6 | CFG_BOOT_EN | BootLoader enable | 1 |
| Bit7 | CFG_ROM_READ | Code and data protection mode in FlashROM: 0-Disable the programmer to read out, and keep the program secret; 1- Read enable | 1 |
| Bit27～bit8 | RESERVED | Reserved | FFF0Fh |
| Bit31～bit28 | VALID_SIG | Configuration information valid flag, fixed value | 0100b |

### 2.3.2 On-chip DATA area mapping table

Table 2-4 DATA area address

| Address range | Application | Description |
|---|---|---|
| 0x20000000-0x200077FF | Independently maintainable memory area supplied by the main + auxiliary dual power, RAM30K | 30KB |
| 0x20007800-0x20007FFF | Independently maintainable memory area supplied by the main + auxiliary dual power, RAM2K | 2KB |

### 2.3.3 Peripheral address assignment

CH583 mainly contains the following peripherals. Each peripheral occupies a certain address space, and the actual access address of peripheral register is: base address + offset address. In the following chapters, the address of the register is described in detail. The following table shows the assignment of base address of each peripheral.

Table 2-5 Peripheral base address assignment table

| Peripheral No. | Peripheral name | Peripheral base address |
|---|---|---|
| 1 | SYS (PMU/RTC/GPIO, etc.)<br>AUX (ADC/TKEY/PLL, etc.) | 0x4000 1000 |
| 2 | FlashROM-Control | 0x4000 1800 |
| 3 | TMR0 | 0x4000 2000 |
| 4 | TMR1 | 0x4000 2400 |
| 5 | TMR2 | 0x4000 2800 |
| 6 | TMR3 | 0x4000 2C00 |
| 7 | UART0 | 0x4000 3000 |
| 8 | UART1 | 0x4000 3400 |
| 9 | UART2 | 0x4000 3800 |
| 10 | UART3 | 0x4000 3C00 |
| 11 | SPI0 | 0x4000 4000 |
| 12 | SPI1 | 0x4000 4400 |
| 13 | I2C | 0x4000 4800 |
| 14 | PWMx (PWM4~PWM11) | 0x4000 5000 |
| 15 | USB | 0x4000 8000 |
| 16 | USB2 | 0x4000 8400 |
| 17 | Radio:BLE | 0x4000 C000<br>0x4000 D000 |

The following table shows the explanation of "Access" in the register description in the subsequent chapters:

Table 2-6 Description of access attributes

| Abbreviation | Description |
|---|---|
| RF | Software can only read this bit. The read value is fixed, not affected by reset. |
| RO | Software can only read these bits. |
| WO | Software can only write to this bit. The read value is 0, or invalid. |
| RZ | Software can only read this bit. Reading this bit automatically clears it to '0'. |
| WZ | Software can clear this bit by writing. |
| RW | Software can read and write to these bits. |
| RW1 | Software can read as well as clear this bit by writing 1. |
| WA | Software can only write, only in safe mode. The read value is 0, or invalid. |
| RWA | Software can read this bit, and can only write to this bit in safe mode. |

The following table explains the abbreviations used in subsequent chapters:

Table 2-7 Description of noun abbreviations

| Abbreviation | Description |
|---|---|
| HSE | External high-frequency crystal oscillator clock source (32MHz recommended) |
| LSE | External low-frequency crystal oscillator clock source (32KHz recommended) |

| | |
|---|---|
| LSI | Internal low-frequency RC clock oscillator source (32KHz after calibration when the application software is running) |
| CK32M | High-frequency clock source (32MHz by default) |
| CK32K | Low-frequency clock source (32KHz by default) |
| Fpll | PLL output clock (480MHz by default) |
| HCLK | System clock |
| Fsys | System clock frequency |
| Tsys | System clock cycle (1/Fsys) |
| RAM2K | 2KB SRAM of high address |
| RAM30K | 30KB SRAM of low address |
| 0x | The data starting with it indicates a hexadecimal number |
| H | The data ending with it indicates a hexadecimal number |
| B | The data ending with it indicates a binary number |

# Chapter 3 Interrupt

The system has a built-in programmable fast interrupt controller (PFIC), which supports up to 255 interrupt vectors. The current system manages 20 peripheral interrupt channels and 8 core interrupt channels, and other interrupt sources are reserved.

## 3.1 Interrupt controller

20 peripheral interrupts; each interrupt request has an independent trigger and maskable control bit, as well as a dedicated status bit.

1 non-maskable interrupt NMI.

Unique fast interrupt entry and exit mechanism, hardware automatic stacking and recovery, without instruction overhead.

Unique fast interrupt response mechanism, 4 channels programmable directly access interrupt vector addresses.

## 3.2 System timer (SysTick)

The core provides a 64-bit counter (SysTick), supports HCLK or HCLK/8 as the time base, with higher priority.

## 3.3 Interrupt and exception vector

The following is the vector table.

Table 3-1 Interrupt vector table

| No. | Priority | Priority type | Name | Description | Address |
|---|---|---|---|---|---|
| 0 | - | - | - | Reserved | 0x0000_0000 |
| 1 | -3 | Fixed | Reset | Reset | 0x0000_0004 |
| 2 | -2 | Fixed | NMI | Non-maskable interrupt | 0x0000_0008 |
| 3 | -1 | Fixed | EXC | Failures and exception interrupts of all types | 0x0000_000C |
| 4 | - | - | - | Reserved | - |
| 5 | -1 | Fixed | ECALL-M | Machine mode callback interrupt | 0x0000_0014 |
| 6-7 | - | - | - | Reserved | - |
| 8 | -1 | Fixed | ECALL-U | User mode callback interrupt | 0x0000_0020 |
| 9 | -1 | Fixed | BREAKPOINT | Breakpoint callback interrupt | 0x0000_0024 |
| 10-11 | - | - | - | Reserved | - |
| 12 | 0 | Settable | SysTick | SysTick timer | 0x0000_0030 |
| 13 | - | - | - | Reserved | - |
| 14 | 1 | Settable | SWI | Software interrupt | 0x0000_0038 |
| 15 | - | - | - | Reserved | 0x0000_003C |

| | | | | | |
|---|---|---|---|---|---|
| 16 | 2 | Settable | TMR0 | TMR0 interrupt | 0x0000_0040 |
| 17 | 3 | Settable | GPIO_A | GPIO port PA interrupt | 0x0000_0044 |
| 18 | 4 | Settable | GPIO_B | GPIO port PB interrupt | 0x0000_0048 |
| 19 | 5 | Settable | SPI0 | SPI0 interrupt | 0x0000_004C |
| 20 | 6 | Settable | BLEL | LLE interrupt of wireless module | 0x0000_0050 |
| 21 | 7 | Settable | BLEB | BB interrupt of wireless module | 0x0000_0054 |
| 22 | 8 | Settable | USB | USB interrupt | 0x0000_0058 |
| 23 | 9 | Settable | USB2 | USB2 interrupt | 0x0000_005C |
| 24 | 10 | Settable | TMR1 | TMR1 interrupt | 0x0000_0060 |
| 25 | 11 | Settable | TMR2 | TMR2 interrupt | 0x0000_0064 |
| 26 | 12 | Settable | UART0 | UART0 interrupt | 0x0000_0068 |
| 27 | 13 | Settable | UART1 | UART1 interrupt | 0x0000_006C |
| 28 | 14 | Settable | RTC | RTC interrupt | 0x0000_0070 |
| 29 | 15 | Settable | ADC | ADC and TouchKey interrupt | 0x0000_0074 |
| 30 | 16 | Settable | I2C | I2C interrupt | 0x0000_0078 |
| 31 | 17 | Settable | PWMX_SPI1 | PWMX (PWM4~11) interrupt and SPI1 interrupt | 0x0000_007C |
| 32 | 18 | Settable | TMR3 | TMR3 interrupt | 0x0000_0080 |
| 33 | 19 | Settable | UART2 | UART2 interrupt | 0x0000_0084 |
| 34 | 20 | Settable | UART3 | UART3 interrupt | 0x0000_0088 |
| 35 | 21 | Settable | WDOG_BAT | Watchdog timer interrupt/battery low voltage interrupt | 0x0000_008C |

## 3.4 Registers

### 3.4.1 PFIC register description

PFIC register base address: 0xE000E000

Table 3-2 PFIC registers

| Name | Offset address | Description | Reset value |
|---|---|---|---|
| R32_PFIC_ISR1 | 0x00 | PFIC interrupt enable status register 1 | 0x00000000 |
| R32_PFIC_ISR2 | 0x04 | PFIC interrupt enable status register 2 | 0x00000000 |
| R32_PFIC_IPR1 | 0x20 | PFIC interrupt pending status register 1 | 0x00000000 |
| R32_PFIC_IPR2 | 0x24 | PFIC interrupt pending status register 2 | 0x00000000 |
| R32_PFIC_ITHRESDR | 0x40 | PFIC interrupt priority threshold configuration register | 0x00000000 |
| R32_PFIC_CFGR | 0x48 | PFIC interrupt configuration register | 0x00000000 |
| R32_PFIC_GISR | 0x4C | PFIC interrupt global status register | 0x00000000 |
| R32_PFIC_IDCFGR | 0x50 | PFIC fast interrupt ID configuration register | 0x00000000 |
| R32_PFIC_FIADDRR0 | 0x60 | PFIC fast interrupt 0 address register | 0x00000000 |
| R32_PFIC_FIADDRR1 | 0x64 | PFIC fast interrupt 1 address register | 0x00000000 |

| | | | |
|---|---|---|---|
| R32_PFIC_FIADDRR2 | 0x68 | PFIC fast interrupt 2 address register | 0x00000000 |
| R32_PFIC_FIADDRR3 | 0x6C | PFIC fast interrupt 3 address register | 0x00000000 |
| R32_PFIC_IENR1 | 0x100 | PFIC interrupt enable register 1 | 0x00000000 |
| R32_PFIC_IENR2 | 0x104 | PFIC interrupt enable register 2 | 0x00000000 |
| R32_PFIC_IRER1 | 0x180 | PFIC interrupt reset enable register 1 | 0x00000000 |
| R32_PFIC_IRER2 | 0x184 | PFIC interrupt reset enable register 2 | 0x00000000 |
| R32_PFIC_IPSR1 | 0x200 | PFIC interrupt pending set register 1 | 0x00000000 |
| R32_PFIC_IPSR2 | 0x204 | PFIC interrupt pending set register 2 | 0x00000000 |
| R32_PFIC_IPRR1 | 0x280 | PFIC interrupt pending reset register 1 | 0x00000000 |
| R32_PFIC_IPRR2 | 0x284 | PFIC interrupt pending reset register 2 | 0x00000000 |
| R32_PFIC_IACTR1 | 0x300 | PFIC interrupt activation status register 1 | 0x00000000 |
| R32_PFIC_IACTR2 | 0x304 | PFIC interrupt activation status register 2 | 0x00000000 |
| R32_PFIC_IPRIORx | 0x400 | PFIC interrupt priority configuration register | 0x00000000 |
| R32_PFIC_SCTLR | 0xD10 | PFIC system control register | 0x00000000 |

In user mode, global interrupt control is supported, please refer to the examples provided in evaluation board documentation.

Description about core interrupt control bit:

1. Reset, NMI, EXC, ECALL-M, ECALL-U and BREAKPOINT interrupts are always enabled by default.
2. NMI and EXC support interrupt suspend clear and set control (controlled by PFIC_IPSR1 and PFIC_IPRR1), but do not support interrupt enable set and clear control.
3. Reset, ECALL-M, ECALL-U and BREAKPOINT do not support interrupt suspend clear and set control, interrupt enable set and clear control.

PFIC interrupt enable status register 1 (PFIC_ISR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | INTENSTA | RO | 31# and below interrupts current enable status.<br>1: Enable the current number interrupt;<br>0: Disable the current number interrupt. | 0 |
| [11:0] | Reserved | RO | Reserved.<br>Reset, NMI, EXC, ECALL and other interrupts bit, the same below. | 0x0C |

PFIC interrupt enable status register 2 (PFIC_ISR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | INTENSTA | RO | 32# and above interrupts current enable status.<br>1: Enable the current number interrupt;<br>0: Disable the current number interrupt. | 0 |

PFIC interrupt suspend status register 1 (PFIC_IPR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [31:12] | PENDSTA | RO | 31# and below interrupts current suspend status.<br>1: Current number interrupt has been suspended;<br>0: Current number interrupt is not suspended. | 0 |
| [11:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt suspend status register 2 (PFIC_IPR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | PENDSTA | RO | 32# and above interrupts current suspend status.<br>1: Current number interrupt has been suspended;<br>0: Current number interrupt is not suspended. | 0 |

PFIC interrupt priority threshold configuration register (PFIC_ITHRESDR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:8] | Reserved | RO | Reserved | 0 |
| [7:0] | THRESHOLD | RW | Interrupt priority threshold setting value.<br>If the interrupt priority value is lower than the current setting value, interrupt service will not be performed when suspended. When this register is 0, the threshold register function is invalid.<br>[7:4]: Priority threshold.<br>[3:0]: Reserved; 0 constantly; invalid if writing. | 0 |

PFIC interrupt configuration register (PFIC_CFGR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | KEYCODE | WO | For different target control bits, the corresponding security access identification data needs to be written synchronously for modification, and the read value is always 0:<br>KEY1 = 0xFA05.<br>KEY2 = 0xBCAF.<br>KEY3 = 0xBEEF. | 0 |
| [15:8] | Reserved | RO | Reserved | 0 |
| 7 | RESETSYS | WO | System reset (Write into KEY3 synchronously). Cleared to 0 automatically.<br>Valid when writing 1; invalid when writing 0.<br>The function of this bit is the same as that of SYSRESET bit of PFIC_SCTLR register. | 0 |
| [6:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt global status register (PFIC_GISR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [31:10] | Reserved | RO | Reserved | 0 |
| 9 | GPENDSTA | RO | Whether there is interrupt pending currently:<br>1: Yes; 0: No. | 0 |
| 8 | GACTSTA | RO | Whether the interrupt is executed currently:<br>1: Yes; 0: No. | 0 |
| [7:0] | NESTSTA | RO | Current interrupt nesting status, support 2-level nesting currently, [1:0] valid.<br>3: Level 2 interrupt in process;<br>1: Level 1 interrupt in process;<br>0: No interrupt occurs;<br>Others: Impossible condition. | 0 |

PFIC fast interrupt ID configuration register (PFIC_IDCFGR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | FIID3 | RW | Configure interrupt number of fast interrupt 3. | 0 |
| [23:16] | FIID2 | RW | Configure interrupt number of fast interrupt 2. | 0 |
| [15:8] | FIID1 | RW | Configure interrupt number of fast interrupt 1. | 0 |
| [7:0] | FIID0 | RW | Configure interrupt number of fast interrupt 0. | 0 |

PFIC fast interrupt 0 address register (PFIC_FIADDRR0)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR0 | RW | Fast interrupt 0 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | FI0EN | RW | Fast interrupt 0 channel enable bit:<br>1: Enable fast interrupt 0 channel; 0: Disable. | 0 |

PFIC fast interrupt 1 address register (PFIC_FIADDRR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR1 | RW | Fast interrupt 1 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | FI1EN | RW | Fast interrupt 1 channel enable bit:<br>1: Enable fast interrupt 1 channel; 0: Disable. | 0 |

PFIC fast interrupt 2 address register (PFIC_FIADDRR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR2 | RW | Fast interrupt 2 service program address bit[31:1], bit0 is 0. | 0 |
| 0 | FI2EN | RW | Fast interrupt 2 channel enable bit:<br>1: Enable fast interrupt 2 channel; 0: Disable. | 0 |

PFIC fast interrupt 3 address register (PFIC_FIADDRR3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | ADDR3 | RW | Fast interrupt 3 service program address bit[31:1], | 0 |

| | | | | |
|---|---|---|---|---|
| | | | bit0 is 0. | |
| 0 | FI3EN | RW | Fast interrupt 3 channel enable bit:<br>1: Enable fast interrupt 3 channel; 0: Disable. | 0 |

PFIC interrupt enable register 1 (PFIC_IENR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | INTEN | WO | 31# and below interrupts enable.<br>1: Enable the current number interrupt;<br>0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt enable register 2 (PFIC_IENR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | INTEN | WO | 32# and above interrupts enable.<br>1: Enable the current number interrupt;<br>0: No effect. | 0 |

PFIC interrupt reset enable register 1 (PFIC_IRER1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | INTRESET | WO | 31# and below interrupts disable.<br>1: Disable the current number interrupt;<br>0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt reset enable register 2 (PFIC_IRER2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | INTRESET | WO | 32# and above interrupts disable.<br>1: Disable the current number interrupt;<br>0: No effect. | 0 |

PFIC interrupt pending set register 1 (PFIC_IPSR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | PENDSET | WO | 31# and below interrupts pending sett.<br>1: Current number interrupt is pending;<br>0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt pending set register 2 (PFIC_IPSR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | PENDSET | WO | 32# and above interrupts pending set. | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Current number interrupt is pending;<br>0: No effect. | |

PFIC interrupt pending reset register 1 (PFIC_IPRR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | PENDRESET | WO | 31# and below interrupts pending reset.<br>1: Current number interrupt reset pending status;<br>0: No effect. | 0 |
| [11:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt pending reset register 2 (PFIC_IPRR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | PENDRESET | WO | 32# and above interrupts pending reset.<br>1: Current number interrupt reset pending status;<br>0: No effect. | 0 |

PFIC interrupt activation status register 1 (PFIC_IACTR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:12] | IACTS | RW1 | 31# and below interrupts activation status.<br>1: Executing the current number interrupt;<br>0: The current number interrupt is not executed. | 0 |
| [11:0] | Reserved | RO | Reserved | 0 |

PFIC interrupt activation status register 2 (PFIC_IACTR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:4] | Reserved | RO | Reserved | 0 |
| [3:0] | IACTS | RW1 | 32# and above interrupts activation status.<br>1: Executing the current number interrupt;<br>0: The current number interrupt is not executed. | 0 |

PFIC interrupt priority configuration register (PFIC_IPRIORx) (x=0-63)

The controller supports 256 interrupts (0-255), and 8 bits are used to set the control priority for each interrupt.

| | 31 24 | 23 16 | 15 8 | 7 0 |
|---|---|---|---|---|
| IPRIOR63 | PRIO_255 | PRIO_254 | PRIO_253 | PRIO_252 |
| … | … | … | … | … |
| IPRIORx | PRIO_(4x+3) | PRIO_(4x+2) | PRIO_(4x+1) | PRIO_(4x) |
| … | … | … | … | … |
| IPRIOR0 | PRIO_3 | PRIO_2 | PRIO_1 | PRIO_0 |

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| [2047:2040] | IP_255 | RW | Same as IP_0 description. | 0 |
| … | … | … | … | … |
| [31:24] | IP_3 | RW | Same as IP_0 description. | 0 |
| [23:16] | IP_2 | RW | Same as IP_0 description. | 0 |
| [15:8] | IP_1 | RW | Same as IP_0 description. | 0 |
| [7:0] | IP_0 | RW | Number 0 interrupt priority configuration:<br>[7:4]: Priority control bit.<br>[3:0]: Reserved. Always 0. Invalid if writing.<br>The smaller priority value means higher priority.<br>Only 2-level nested interrupts, i.e., it can be only preempted once. | 0 |

PFIC system control register (PFIC_SCTLR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | SYSRESET | WO | System reset. Cleared to 0 automatically. Valid when writing 1, while invalid when writing 0.<br>The same effect as PFIC_CFGR register. | 0 |
| [30:6] | Reserved | RO | Reserved | 0 |
| 5 | SETEVENT | WO | Set event to wake up the WFE. | 0 |
| 4 | SEVONPEND | RW | When an event or interrupt suspending status occurs, the system can be woken up by the WFE command. If the WFE command is not executed, the system will be woken up immediately after the next execution of the command.<br>1: Enable events and all interrupts (including disabled interrupts) can wake up the system;<br>0: Only enabled events and enabled interrupts can wake up the system. | 0 |
| 3 | WFITOWFE | RW | The WFI command is executed as WFE.<br>1: The subsequent WFI command is deemed as WFE command;<br>0: No action. | 0 |
| 2 | SLEEPDEEP | RW | Low power mode of control system:<br>1: deepsleep 0: sleep | 0 |
| 1 | SLEEPONEXIT | RW | The system status after controlled to exit the interrupt service program:<br>1: The system gets into low-power mode;<br>0: The system gets into the main program. | 0 |
| 0 | Reserved | RO | Reserved | 0 |

### 3.4.2 CSR registers defined by WCH

In RISC-V, some control and status registers (CSR) are defined， which are used to configure, mark and record run status. CSR registers belong to internal registers of the core, and have a dedicated 12-bit address apace. WCH devices not only provide standard registers defined in RISC-V architecture documentation, but also provide some

registers defined by manufactures and the csr instruction is needed to access. Note: These registers with "MRW" attribute can only be accessed when the system is in machine mode.

Interrupt system control register (INTSYSCR)

CSR address: 0x804

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:2] | Reserved | RO | Reserved | 0 |
| 1 | INESTEN | MRW | Interrupt nesting enable.<br>1: Enable; 0: Disable. | 0 |
| 0 | HWSTKEN | MRW | Hardware stack function enable<br>1: Enable; 0: Disable. | 0 |

Machine trap-vector base-address register (MTVEC)

CSR address: 0x305

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:2] | BASEADD | RO | Interrupt vector table base address. | 0 |
| 1 | MODE1 | RW | Interrupt vector table recognize mode.<br>1: Recognize based on absolute address, support full range, but must jump;<br>0: Recognize based on jump instruction, limited range, support non-jump instruction. | 0 |
| 0 | MODE0 | RW | Interrupt or exception entry address mode select:<br>1: Offset address based on number*4;<br>0: Unified entry address. | 0 |

### 3.4.3 Physical memory protection (PMP)

In order to improve system security, a set of physical address access privileges are defined in the RISC-V architecture, which can be used to set read/write/execute attribute of physical memory in the space and support regions as small as 4 bytes. PMP unit is always effective in user mode, and is optionally effective in machine mode. If the current memory restriction is violated, it may cause system exception interrupt (EXC).

PMP unit contains 4 sets of 8-bit configuration registers (in total 32 bits) and 4 sets of address registers, which need csr instruction to access and the registers must be in machine mode.

PMP configuration register (PMPCFG0)

CSR address: 0x3A0

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | pmp3cfg | MRW | See pmp0cfg. | 0 |
| [23:16] | pmp2cfg | MRW | See pmp0cfg. | 0 |
| [15:8] | pmp1cfg | MRW | See pmp0cfg. | 0 |

<table>
<tr><td>[7:0]</td><td>pmp0cfg</td><td>MRW</td><td>
<table>
<tr><th>Bit</th><th>Name</th><th>Description</th></tr>
<tr><td>7</td><td>L</td><td>Lock enable. Disable lock in machine mode:<br>1: Lock related register;<br>0: Not lock.</td></tr>
<tr><td>[6:5]</td><td>-</td><td>Reserved</td></tr>
<tr><td>[4:3]</td><td>A</td><td>Address alignment and protection region range selection.</td></tr>
<tr><td>2</td><td>X</td><td>Execute.</td></tr>
<tr><td>1</td><td>W</td><td>Write.</td></tr>
<tr><td>0</td><td>R</td><td>Read.</td></tr>
</table>
</td><td>0</td></tr>
</table>

Address alignment and protection region range select, memory protection for the region, A_ADDR ≤ region < B_ADDR (A_ADDR and B_ADDR are required to be 4 bytes aligned):

1. If B_ADDR - A_ADDR == $2^2$, select NA4;
2. If B_ADDR - A_ADDR == $2^{(G+2)}$, G ≥ 1, and A_ADDR is $2^{(G+2)}$ aligned, select NAPOT;
3. Otherwise select TOR.

| A | Name | Description |
|---|---|---|
| 0 | OFF | Null region |
| 1 | TOR | Top of range:<br>pmp0cfg, 0 ≤ region ＜ pmpaddr0;<br>pmp1cfg, pmpaddr0 ≤ region ＜ pmpaddr1;<br>pmp2cfg, pmpaddr1 ≤ region ＜ pmpaddr2;<br>pmp3cfg, pmpaddr2 ≤ region ＜ pmpaddr3.<br>$pmpaddr_{i-1}$ = A_ADDR >> 2;<br>$pmpaddr_i$ = B_ADDR >> 2。 |
| 2 | NA4 | Naturally aligned 4-byte region.<br>pmp0cfg～pmp3cfg correspond to pmpaddr0~pmpaddr3 as start address.<br>$pmpaddr_i$ = A_ADDR >> 2. |
| 3 | NAPOT | $2^{(G+2)}$ region, G≥1, and A_ADDR is $2^{(G+2)}$ aligned.<br>$pmpaddr_i$ = ( A_ADDR >> 2 ) \| ( $2^{(G-1)}$ – 1 ). |

PMP address 0 register (PMPADDR0)

CSR address: 0x3B0

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR0 | MRW | Bit[33:2] of PMP address 0. Actually, the higher 2 bits are not used. | 0 |

PMP address 1 register (PMPADDR1)

CSR address: 0x3B1

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR1 | MRW | Bit[33:2] of PMP set address 1. Actually, the higher 2 bits are not used. | 0 |

PMP address 2 register (PMPADDR2)

CSR address: 0x3B2

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR2 | MRW | Bit[33:2] of PMP set address 2. Actually, the higher 2 bits are not used. | 0 |

PMP address 3 register (PMPADDR3)

CSR address: 0x3B3

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | ADDR3 | MRW | Bit[33:2] of PMP set address 3. Actually, the higher 2 bits are not used. | 0 |

### 3.4.4 SysTick register description

STK register base address: 0xE000F000

| Name | Offset address | Description | Reset value |
|---|---|---|---|
| R32_STK_CTRL | 0x00 | System count control register | 0x00000000 |
| R32_STK_SR | 0x04 | System count status register | 0x00000000 |
| R32_STK_CNTL | 0x08 | System counter low register | 0x00000000 |
| R32_STK_CNTH | 0x0C | System counter high register | 0x00000000 |
| R32_STK_CMPLR | 0x10 | Count reload low register | 0x00000000 |
| R32_STK_CMPHR | 0x14 | Count reload high register | 0x00000000 |

System count control register (STK_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 31 | SWIE | RW | Software interrupt trigger enable (SWI):<br>1: Trigger software interrupt;<br>0: Disable trigger. This bit must be cleared after going into software interrupt, otherwise trigger will be always enabled. | 0 |
| [30:6] | Reserved | RO | Reserved | 0 |
| 5 | INIT | W1 | Counter initial value update:<br>1: Upcount from 0, downcount from comparison value;<br>0: No effect. | 0 |
| 4 | MODE | RW | Count mode:<br>1: Downcount; 0: Upcount. | 0 |
| 3 | STRE | RW | Auto reload count enable:<br>1: Count from 0 after upcounting to comparison value, downcount from comparison value after downcounting to 0;<br>0: Continue upcounting/downcounting. | 0 |
| 2 | STCLK | RW | Counter clock source select:<br>1: HCLK as timebase;<br>0: HCLK/8 as timebase. | 0 |
| 1 | STIE | RW | Counter interrupt enable control bit: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Enable counter interrupt; 0: No counter interrupt. | |
| 0 | STE | RW | System counter enable control bit:<br>1: Enable system counter STK;<br>0: Disable system counter STK; the counter stops counting. | 0 |

System count status register (STK_SR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:1] | Reserved | RO | Reserved | 0 |
| 0 | CNTIF | RW0 | Count value compare flag, cleared when writing 0, while invalid when writing 1:<br>1: Upcount to comparison value, downcount to 0;<br>0: Not reach the comparison value. | 0 |

System counter low register (STK_CNTL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CNTL | RW | Lower 32 bits of STK counter count value. | 0 |

System counter high register (STK_CNTH)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CNTH | RW | Higher 32 bits of STK counter count value. | 0 |

Count reload low register (STK_CMPLR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CMPL | RW | Set reload counter value lower 32 bits. | 0 |

Count reload high register (STK_CMPHR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | CMPH | RW | Set reload counter value higher 32 bits. | 0 |

# Chapter 4 System Control

## 4.1 Reset control

The system supports 6 types of resets, including Real Power on Reset (RPOR), external Manual Reset (MR), internal Software Reset (SR), Watch-dog Time-out Reset (WTR), Global Reset by Waking under Shutdown Mode (GRWSM) and Local Reset by Waking (LRW).

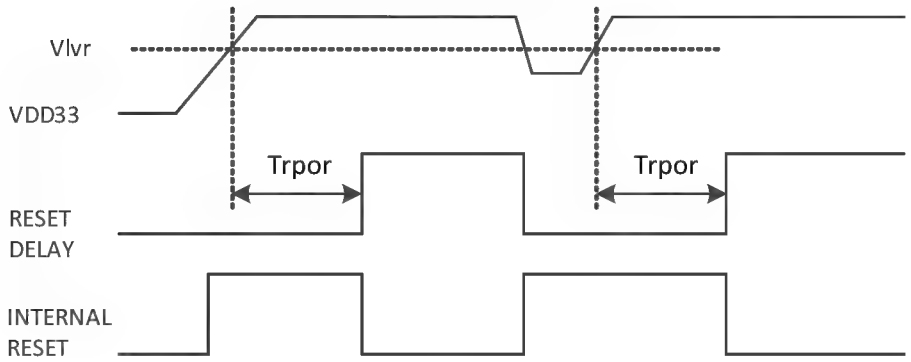
The R8_GLOB_RESET_KEEP register and the RB_ROM_CODE_OFS register are reset only when RPOR or GRWSM occurs, and are not affected by other types of resets.

Please refer to the timing parameter table in Section 20.5 for the timing parameters and reset property parameters in the figure below.

### 4.1.1 Real Power On Reset (RPOR)

When the power is on, the POR module inside the chip will generate a power-on reset and delay to wait for the power to stabilize. In addition, during operation, when the power voltage is lower than Vlvr, the internal LVR module of the chip will generate a low voltage reset until the voltage rises, and delay to wait for the power to stabilize. The figure below shows the power-on reset process and the low-voltage reset process.

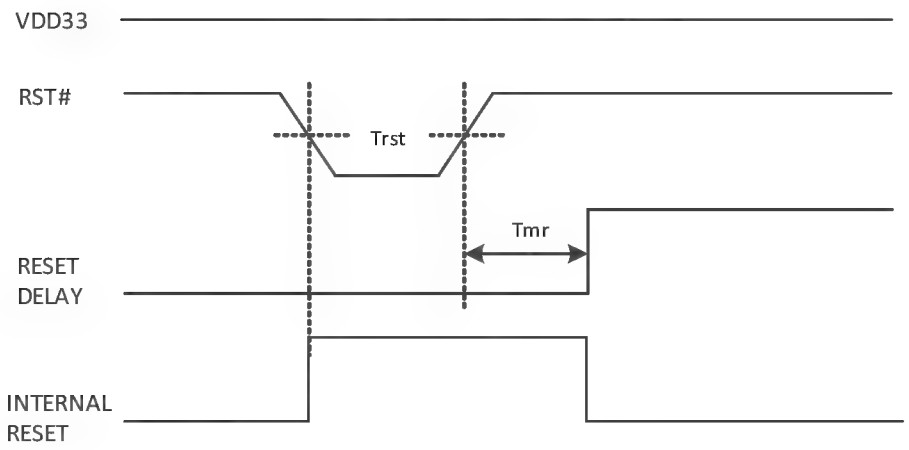
Figure 4-1 Real power on reset




### 4.1.2 External Manual Reset (MR)

The external manual reset is triggered by a low level externally added to the RST# pin. When the duration of reset low level is greater than the minimum reset pulse width (Trst), the system is triggered to reset.

Figure 4-2 External manual reset




### 4.1.3 Internal Software Reset (SR)

Internal software reset is automatically carried out without external intervention. Set the bit RB_SOFTWARE_RESET of global reset configuration register (R8_RST_WDOG_CTRL) to 1, to realize software reset. This bit will be automatically cleared to 0.

### 4.1.4 Watchdog Time-out Reset (WTR)

Watchdog function is based on an 8-bit count-up counter with a count clock cycle of 131072/Fsys. When the watchdog timeout reset function is turned on, the entire system will be reset once this counter overflows.

### 4.1.5 Global Reset by Waking under Shutdown Mode (GRWSM)

Once the system enters the shutdown mode (see the power management chapter for details), the system will perform the wake-up operation in an orderly manner under the action of the wake-up signal, and the system will perform a global reset after wake-up. This reset effect is similar to that of power-on reset.

### 4.1.6 Local Reset by Waking (LRW)

If the system is awakened from sleep mode, a reset will be generated after the associated power is ready. It is a partial reset, with a selective reset of the registers that are powered down in sleep mode as needed.

In sleep mode, the registers of each functional module are divided into 3 categories:

The first type is the key registers belonging to the functional module that requires data retention (such as configuration/mode, etc.). At the time of sleep, the auxiliary power continues to supply power, and the data is not lost, both sleep and wake-up have no effect on data;

The second type is the regenerative registers belonging to the functional module that requires data retention (such as counters, FIFOs, etc.). The power is turned off during sleep, and the data is a random number (such as FIFO memory cell) or reset (such as FIFO counter) after waking up;

The third type is the register belonging to the functional module that does not require data retention. The power is turned off during sleep, and the data is a random number (such as FIFO memory cell) or reset (such as FIFO counter, configuration/mode register) after waking up.

LRW is used for the latter 2 reset registers above.

## 4.2 Safe access

The attributes of some registers of the system are "RWA" or "WA", indicating that the current register can be safely accessed and can be read directly, but needs to enter the safe access mode when write-in:

First write 0x57 to the R8_SAFE_ACCESS_SIG register;

Then write 0xA8 to the R8_SAFE_ACCESS_SIG register;

At this time, you can enter the safe access mode and operate the registers with the attribute "RWA/WA". After that, about 16 system frequency cycles (Tsys) are in safe mode, and one or more secure registers can be rewritten within the valid period. The safe mode will be automatically terminated after the above validity period is exceeded. Or you can write 0x00 in the R8_SAFE_ACCESS_SIG register in advance to terminate the safe mode.

## 4.3 Register description

Table 4-1 System control registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_SAFE_ACCESS_SIG | 0x40001040 | Safe access flag register | 0x00 |
| R8_CHIP_ID | 0x40001041 | Chip ID register | 0x83 |
| R8_SAFE_ACCESS_ID | 0x40001042 | Safe access ID register | 0x0C |
| R8_WDOG_COUNT | 0x40001043 | Watchdog counter register | 0x00 |
| R8_RESET_STATUS | 0x40001044 | Reset status register | 0x01 |
| R8_GLOB_ROM_CFG | 0x40001044 | FlashROM application configuration register | 0x01 |
| R8_GLOB_CFG_INFO | 0x40001045 | Global configuration information status register | 0xEX |
| R8_RST_WDOG_CTRL | 0x40001046 | Watchdog and reset configuration register | 0x00 |
| R8_GLOB_RESET_KEEP | 0x40001047 | Reset keep register | 0x00 |
| R32_FLASH_DATA | 0x40001800 | FlashROM word data register | 0xXXXXXXXX |
| R32_FLASH_CONTROL | 0x40001804 | FlashROM control register | 0x074000XX |
| R8_FLASH_DATA | 0x40001804 | FlashROM byte data register | 0xXX |
| R8_FLASH_CTRL | 0x40001806 | FlashROM access control register | 0x40 |
| R8_FLASH_CFG | 0x40001807 | FlashROM access configuration register | 0x07 |

Safe access flag register (R8_SAFE_ACCESS_SIG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_SAFE_ACCESS_SIG | WO | Safe access flag register.<br>Some registers (access attribute is RWA) are protection registers and write operation can be conducted only after entering the safe access mode. Write 0x57 first and then 0xA8 to this register, to enter the safe access mode. The time is limited to about 16 main clock cycles (Tsys), and automatic protection is enabled if it is exceeded. You can write | 00h |

| | | | | |
|---|---|---|---|---|
| | | | any other value to force direct exit from the safe access mode and return to the protected state. | |
| 7 | Reserved | R0 | Reserved | 0 |
| [6:4] | RB_SAFE_ACC_TIMER | RO | Count the current safe access time. | 000b |
| 3 | RB_SAFE_ACC_ACT | RO | Current safe access mode status:<br>1: Writable in unlocked/safe access mode;<br>0: Locked, register with RWA attribute cannot be rewritten. | 0 |
| 2 | Reserved | R0 | Reserved | 0 |
| [1:0] | RB_SAFE_ACC_MODE | RO | Current safe access mode status.<br>11: Safe mode, register with RWA attribute can be written;<br>Others: Non-safe mode. | 00b |

Chip ID register (R8_CHIP_ID)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_CHIP_ID | RF | CH583: Always 83h, to identify the chip.<br>CH582: Always 82h, to identify the chip. | 83h<br>82h |

Safe access ID register (R8_SAFE_ACCESS_ID)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_SAFE_ACCESS_ID | RF | Fixed value 0Ch. | 0Ch |

Watchdog counter register (R8_WDOG_COUNT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_WDOG_COUNT | RW | Watchdog counter, whose initial value can be preset, is always automatically incremented, and can be cycled from 0xFF to 0x00 and then continue.<br>Count cycle = 131072/Fsys. | 00h |

Reset status register (R8_RESET_STATUS)/ FlashROM application configuration register (R8_GLOB_ROM_CFG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_ROM_CODE_WE | RWA | Erase/program enable bit of CodeFlash in FlashROM program memory area:<br>1: Erase/program is enabled;<br>0: Erase/program protection for this area. | 0 |
| 6 | RB_ROM_DATA_WE | RWA | Erase/program enable bit of DataFlash in FlashROM data memory area:<br>1: Erase/program is enabled;<br>0: Erase/program protection for all. | 0 |
| 5 | RB_ROM_CTRL_EN | RWA | FlashROM access control interface enable: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Control enabled;<br>0: Access disabled. | |
| 4 | RB_ROM_CODE_OFS | RWA | Select the start offset address of user program code in FlashROM. This value is not affected by MR, SR, WTR or GRWSM, and can be cleared only when RPOR is valid:<br>0: 0x000000;<br>1: 0x040000 (skip the first 256KB in ROM). | 0 |
| 3 | Reserved | R0 | Reserved | 0 |
| [2:0] | RB_RESET_FLAG | RO | Last reset state:<br>000: Software reset SR (generated when RB_WDOG_RST_EN=0 and software reset, otherwise, it can be reset but this status is not generated);<br>001: Real Power-on reset (RPOR);<br>010: Watchdog timeout reset (WTR);<br>011: External manual reset (MR);<br>101: Global Reset by Waking under Shutdown Mode (GRWSM)<br>100/110/111: Local Reset by waking (LRW), and the last reset is SR/WTR/MR. | 001b |

Global configuration information status register (R8_GLOB_CFG_INFO)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：6] | Reserved | RO | Reserved | 11b |
| 5 | RB_BOOT_LOADER | RO | Bootloader status:<br>1: Currently in Bootloader status;<br>0: Currently in application program status. | 1/0 |
| 4 | RB_CFG_DEBUG_EN | RO | 2-wire simulation debug interface enable:<br>1: Enable simulation and debug, and FlashROM can be read;<br>0: Disable simulation and debug. | 0 |
| 3 | RB_CFG_BOOT_EN | RO | System BootLoader enable status:<br>1: Enable;<br>0: Disable. | 1 |
| 2 | RB_CFG_RESET_EN | RO | RST# external manual reset enable status:<br>1: Enable;<br>0: Disable. | 0 |
| 1 | Reserved | RO | Reserved | 0 |
| 0 | RB_CFG_ROM_READ | RO | Code and data area protection status in FlashROM:<br>1: External programmer is readable;<br>0: Protected, externally inaccessible, and the program is kept secret. | 0 |

Watchdog and reset configuration register (R8_RST_WDOG_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:5] | Reserved | RO | Reserved | 000b |
| 4 | RB_WDOG_INT_FLAG | RW1 | Watchdog timer interrupt flag:<br>1: Watchdog count overflows, that is, R8_WDOG_COUNT is detected to progressively increase from 0xFF to 0x00;<br>0: Watchdog count has not overflowed. Write 1 to clear, or reload the watchdog counter value (R8_WDOG_COUNT) to clear, or execute SEV() to clear. | 0 |
| 3 | Reserved | RO | Reserved | 0 |
| 2 | RB_WDOG_INT_EN | RWA | Watchdog timer interrupt enable bit:<br>1: Enable, an interrupt will be generated after the watchdog count overflows;<br>0: Disable the watchdog timer interrupt. | 0 |
| 1 | RB_WDOG_RST_EN | RWA | Watchdog timeout reset enable bit:<br>1: Enable, system is reset after the watchdog count overflows;<br>0: Only used as watchdog timer.<br>*Note: After this bit is set to 1, the software reset operation will not affect the RB_RESET_FLAG status.* | 0 |
| 0 | RB_SOFTWARE_RESET | WA/ WZ | System software reset control; automatically cleared after reset:<br>1: Perform system software reset;<br>0: Idle, no action. | 0 |

Reset keep register (R8_GLOB_RESET_KEEP)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_GLOB_RESET_KEEP | RW | Reset keep register. The value of this register is not affected by manual reset, software reset, watchdog reset or ordinary wake-up reset. | 00h |

For the operation or setting of FlashROM, please refer to related subprograms. This datasheet does not provide the introductions to FlashROM word data registers and FlashROM control registers.

## 4.4 Flash-ROM operation steps

1. Erase Flash-ROM, please refer to and call related subprograms.
2. Write Flash-ROM, please refer to and call the related subprograms.
3. Read Flash-ROM, read the code or data of the target address through the pointer to the program memory space.

## 4.5 Unique ID

Each chip has a unique ID (chip identification number) when it is delivered from the factory. The ID data and its checksum are 8 bytes in total, stored in the read-only area of chip. Please refer to the routines for details.

# Chapter 5 Power Control

## 5.1 Power management

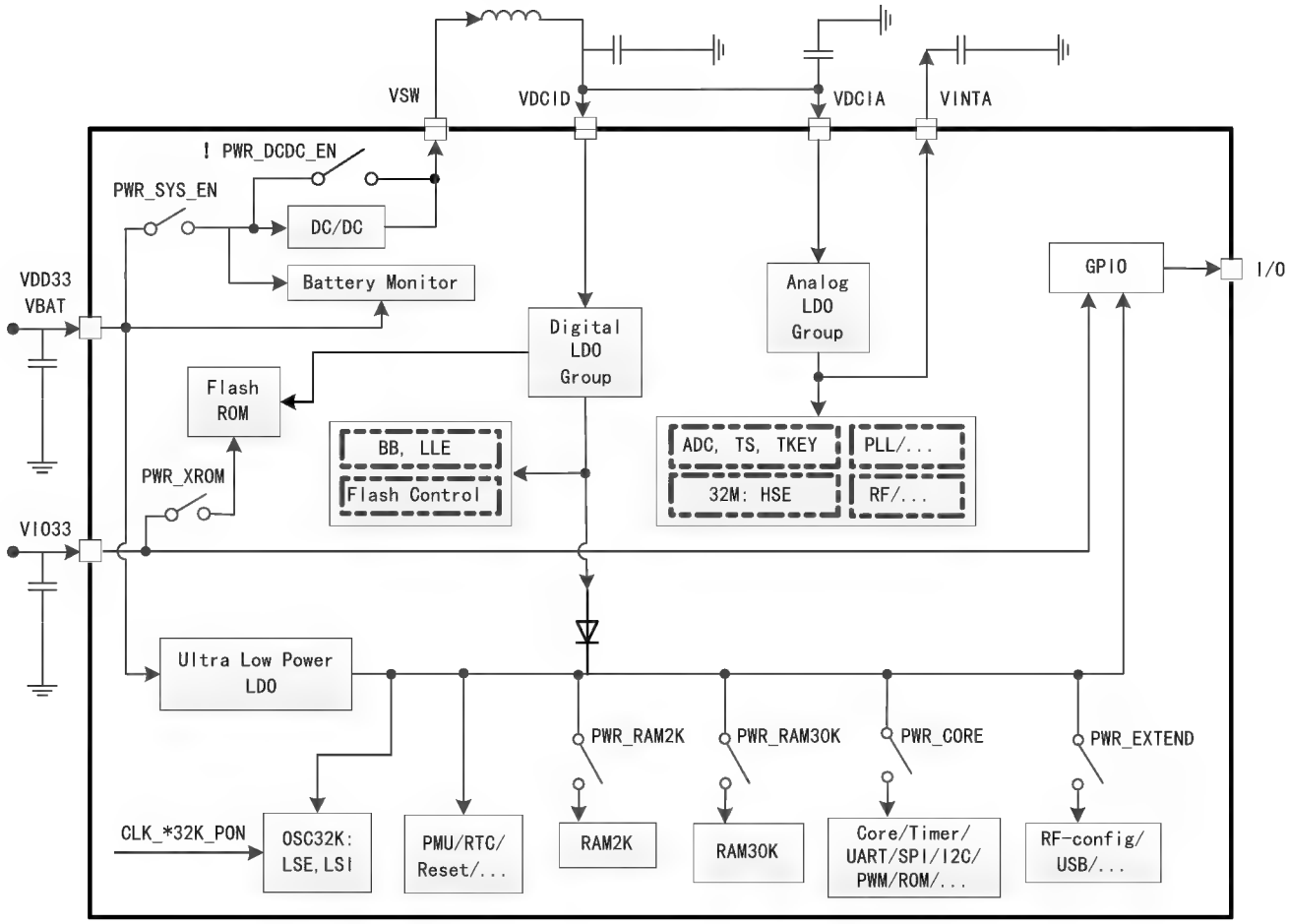
CH583 has a built-in Power Management Unit (PMU). The system power is input from VDD33 to provide the FlashROM, digital circuits (including core, USB, etc.) and analog circuits (including high-frequency oscillator, PLL, ADC and RF transceiver) of system with the required power through the built-in multiple LDO voltage regulators. The power of GPIO and FlashROM is input from VIO33.

There are 2 types of power supply during normal work: direct power and DC-DC conversion. In addition to normal operation, CH583 provides 4 low power modes: idle mode, suspend mode, sleep mode and power-off mode.

DC-DC is not enabled by default after power-on, but a direct power is provided, with a small voltage ripple. In order to reduce the system power consumption during normal operation, you can choose to enable DC-DC to increase the power consumption utilization rate, and the operating current will usually drop to about 60% of that when direct power is used.

In order to reduce the power consumption of the system during sleep, you can choose to turn off the main LDO of the system and switch to the built-in ultra-low power ULP-LDO of the system to provide the auxiliary power. When the system gets into sleep or power-down mode, in addition to power management and RTC registers and other normal power units, you can select whether to maintain the power supply of the high 2KB and low 30KB SRAM, core and all peripherals of system, and whether to enable LSE/LSI.

Figure 5-1 Power system




## 5.2 Register description

Table 5-1 Power management registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_SLP_CLK_OFF | 0x4000100C | Sleep clock control register | 0x0000 |
| R8_SLP_CLK_OFF0 | 0x4000100C | Sleep clock control register 0 | 0x00 |
| R8_SLP_CLK_OFF1 | 0x4000100D | Sleep clock control register 1 | 0x00 |
| R8_SLP_WAKE_CTRL | 0x4000100E | Wake-up event configuration register | 0x20 |
| R8_SLP_POWER_CTRL | 0x4000100F | Peripheral sleep power control register | 0x00 |
| R16_POWER_PLAN | 0x40001020 | Sleep power management register | 0x11DF |
| R16_AUX_POWER_ADJ | 0x40001022 | Auxiliary power adjustment control register | 0x0XXX |
| R8_BAT_DET_CTRL | 0x40001024 | Battery voltage detection control register | 0x00 |
| R8_BAT_DET_CFG | 0x40001025 | Battery voltage detection configuration register | 0x02 |
| R8_BAT_STATUS | 0x40001026 | Battery status register | 0x00 |

Sleep clock control register 0 (R8_SLP_CLK_OFF0)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_SLP_CLK_UART3 | RWA | Clock source of UART3:<br>1: Disable; 0: Enable. | 0 |
| 6 | RB_SLP_CLK_UART2 | RWA | Clock source of UART2:<br>1: Disable; 0: Enable. | 0 |
| 5 | RB_SLP_CLK_UART1 | RWA | Clock source of UART1:<br>1: Disable; 0: Enable. | 0 |
| 4 | RB_SLP_CLK_UART0 | RWA | Clock source of UART0:<br>1: Disable; 0: Enable. | 0 |
| 3 | RB_SLP_CLK_TMR3 | RWA | Clock source of timer 3:<br>1: Disable; 0: Enable. | 0 |
| 2 | RB_SLP_CLK_TMR2 | RWA | Clock source of timer 2:<br>1: Disable; 0: Enable. | 0 |
| 1 | RB_SLP_CLK_TMR1 | RWA | Clock source of timer 1:<br>1: Disable; 0: Enable. | 0 |
| 0 | RB_SLP_CLK_TMR0 | RWA | Clock source of timer 0:<br>1: Disable; 0: Enable. | 0 |

Sleep clock control register 1 (R8_SLP_CLK_OFF1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_SLP_CLK_BLE | RWA | Clock source of BLE controller:<br>1: Disable; 0: Enable. | 0 |
| 6 | Reserved | RO | Reserved | 0 |
| 5 | RB_SLP_CLK_USB2 | RWA | Clock source of USB2 controller:<br>1: Disable; 0: Enable. | 0 |
| 4 | RB_SLP_CLK_USB | RWA | Clock source of USB controller:<br>1: Disable; 0: Enable. | 0 |
| 3 | RB_SLP_CLK_I2C | RWA | I2C clock source:<br>1: Disable; 0: Enable. | 0 |
| 2 | RB_SLP_CLK_PWMX | RWA | PWMx clock source:<br>1: Disable; 0: Enable. | 0 |
| 1 | RB_SLP_CLK_SPI1 | RWA | SPI1 clock source:<br>1: Disable; 0: Enable. | 0 |
| 0 | RB_SLP_CLK_SPI0 | RWA | SPI0 clock source:<br>1: Disable; 0: Enable. | 0 |

Wake-up event configuration register (R8_SLP_WAKE_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| 6 | RB_WAKE_EV_MODE | RWA | Wake-up event internal memory mode enable:<br>1: Enable memory, support short-pulse event wake-up;<br>0: Disable memory, the event should remain | 0 |

| | | | | |
|---|---|---|---|---|
| | | | valid until wake up. | |
| 5 | RB_SLP_BAT_WAKE | RWA | Enable battery low voltage event wake up system:<br>1: Enable; 0: Disable. | 1 |
| 4 | RB_SLP_GPIO_WAKE | RWA | Enable GPIO event wake up system:<br>1: Enable; 0: Disable. | 0 |
| 3 | RB_SLP_RTC_WAKE | RWA | Enable RTC event wake up system:<br>1: Enable; 0: Disable. | 0 |
| 2 | Reserved | RO | Reserved | 0 |
| 1 | RB_SLP_USB2_WAKE | RWA | Enable USB2 event wake up system:<br>1: Enable; 0: Disable. | 0 |
| 0 | RB_SLP_USB_WAKE | RWA | Enable USB event wake up system:<br>1: Enable; 0: Disable. | 0 |

Peripheral sleep power control register (R8_SLP_POWER_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| 6 | RB_RAM_RET_LV | RWA | Auxiliary power low voltage enabled during SRAM sleep:<br>0: Normal power voltage, slightly high-power consumption during sleep;<br>1: Low power voltage, slightly low-power consumption during sleep. | 0 |
| 5 | RB_SLP_CLK_RAM2K | RWA | SRAM clock control of RAM2K:<br>1: Disable; 0: Enable. | 0 |
| 4 | RB_SLP_CLK_RAMX | RWA | Clock control of main SRAM (RAM30K):<br>1: Disable; 0: Enable. | 0 |
| [3:2] | Reserved | RO | Reserved | 00b |
| [1:0] | RB_WAKE_DLY_MOD | RWA | Delay cycle select after wake-up:<br>11: No delay, 8 cycles +TSUCLK, disabled;<br>10: Ultra short delay, 70 cycles +TSUCLK;<br>01: Short delay, 520 cycles +TSUCLK, recommended;<br>00: Long delay, 3590 cycles +TSUCLK.<br>TSUCLK is determined by sleep mode and clock configuration, it may contain startup time of TSUHSE or PLL or the 2, refer to evaluation board examples for specified combinations. | 00b |

Sleep power management register (R16_POWER_PLAN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | RB_PWR_PLAN_EN | RWA/WZ | Sleep power planning control enable:<br>1: Enable planning; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Disable or end planning.<br>The power planning is enabled for execution when entering sleep or power-off mode later, and this bit is automatically cleared after execution. | |
| [14:11] | RB_PWR_MUST_0010 | RWA | Reserved, 0010b must be written. | 0010b |
| 10 | RB_PWR_DCDC_PRE | RWA | DC-DC bias circuit enable (effective immediately):<br>1: Enable; 0: Disable. | 0 |
| 9 | RB_PWR_DCDC_EN | RWA | DC-DC enable bit (effective immediately):<br>1: Enable DC-DC, the direct power is off;<br>0: Disable DC-DC, the direct power is on. | 0 |
| 8 | RB_PWR_LDO_EN | RWA | Internal LDO control (sleep planning):<br>1: Turn on LDO;<br>0: Plan to turn off LDO, saving more power. | 1 |
| 7 | RB_PWR_SYS_EN | RWA | System power control (sleep planning):<br>1: Provide system power (on VSW pin);<br>0: Turn off the system power, plan to enter sleep mode or power-off mode. | 1 |
| 6 | Reserved | RWA | Reserved, 0 must be written. | 1 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_PWR_RAM30K | RWA | SRAM power supply of RAM30K (sleep planning):<br>1: Dual power; 0: No auxiliary power. | 1 |
| 3 | RB_PWR_EXTEND | RWA | USB and RF configuration power supply (sleep planning):<br>1: Dual power; 0: No auxiliary power. | 1 |
| 2 | RB_PWR_CORE | RWA | Power of the core and basic peripherals (sleep planning):<br>1: Dual power; 0: No auxiliary power. | 1 |
| 1 | RB_PWR_RAM2K | RWA | SRAM power supply of RAM2K (sleep planning):<br>1: Dual power; 0: No auxiliary power. | 1 |
| 0 | RB_PWR_XROM | RWA | FlashROM power supply (sleep planning):<br>1: Continuous power;<br>0: Power off during sleep. | 1 |

This register is preset for sleep planning except RB_PWR_DCDC_PRE and RB_PWR_DCDC_EN, and its power configuration will take effect after entering the low-power sleep mode and power-down mode.

Auxiliary power adjustment control register (R16_AUX_POWER_ADJ)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:12] | Reserved | RO | Reserved | 0000b |
| [11:8] | Reserved | RO | Reserved, read-only. Write operation has no | XXXXb |

| | | | | |
|---|---|---|---|---|
| | | | effect. | |
| 7 | RB_DCDC_CHARGE | RWA | Low-power auxiliary DC-DC enable bit:<br>1: Auxiliary DC-DC enabled;<br>0: Auxiliary DC-DC disabled. | 0 |
| 6 | Reserved | RO | Reserved | 0 |
| [5:3] | Reserved | RWA | Reserved, the original value must be kept unchanged when writing. | 1XXb |
| [2:0] | RB_ULPLDO_ADJ | RWA | The auxiliary power output voltage adjustment value of ultra-low power LDO (the value is for reference only, and it is not recommended to modify):<br>000: 0.77V; 001: 0.80V;<br>010: 0.84V; 011: 0.88V;<br>100: 0.91V; 101: 0.95V;<br>110: 0.99V; 111: 1.03V. | XXXb<br>(010b) |

Battery voltage detection control register (R8_BAT_DET_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:4] | Reserved | RO | Reserved | 0000b |
| 3 | RB_BAT_LOW_IE | RWA | Battery low voltage interrupt enable:<br>1: Enable; 0: Disable. | 0 |
| 2 | RB_BAT_LOWER_IE | RWA | Battery ultra-low voltage interrupt enable:<br>1: Enable; 0: Disable. | 0 |
| 1 | RB_BAT_MON_EN | RWA | Low-power battery voltage monitor function enable:<br>1: Enable, increasing about 1uA current;<br>0: Disable. | 0 |
| 0 | RB_BAT_LOW_VTHX | RWA | When RB_BAT_MON_EN=1, set the low-power low voltage monitoring threshold. Refer to RB_BAT_LOW_VTH. | 0 |
| 0 | RB_BAT_DET_EN | RWA | High-precision battery voltage detection function enable when RB_BAT_MON_EN=0<br>1: Enable, and turn on modules such as reference voltage simultaneously, and the current is 210uA in sleep mode;<br>0: Disable. | 0 |

*Note: If the battery voltage reaches the lower voltage detection threshold and both RB_BAT_LOWER_IE and RB_BAT_LOW_IE are enabled (only one of them is enabled under normal conditions), NMI non-maskable interrupt will be generated, which is equivalent to increasing the interrupt priority.*

Battery voltage detection configuration register (R8_BAT_DET_CFG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:2] | Reserved | RO | Reserved | 000000b |
| [1:0] | RB_BAT_LOW_VTH | RWA | When RB_BAT_MON_EN=0, set the high- | 10b |

| | | | | |
|---|---|---|---|---|
| | | | precision ultra-low voltage and low voltage detection threshold: (ultra-low voltage reference threshold, low voltage reference threshold)<br>00: 1.7V , 1.95V;<br>01: 1.9V , 2.15V;<br>10: 2.1V , 2.35V;<br>11: 2.3V , 2.55V.<br>When RB_BAT_MON_EN=1, take RB_BAT_LOW_VTHX as the highest bit, adding these 2 bits, these 3 bits are used to set the low-power low-voltage monitoring threshold:<br>000: 1.7V; 001: 1.8V;<br>010: 1.9V; 011: 2.0V;<br>100: 2.1V; 101: 2.2V;<br>110: 2.3V; 111: 2.4V. | |

Battery status register (R8_BAT_STATUS)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:2] | Reserved | RO | Reserved | 000000b |
| 1 | RB_BAT_STAT_LOW | RO | The result of battery low-voltage detection or low-voltage monitoring, indicating that the battery voltage is in low voltage status:<br>1: Below the low voltage threshold; 0: No. | 0 |
| 0 | RB_BAT_STAT_LOWER | RO | When RB_BAT_MON_EN=0, it indicates that battery voltage is in ultra-low voltage status:<br>1: Below the ultra-low voltage threshold;<br>0: No. | 0 |

## 5.3 Low power mode

After the system is reset, the microcontroller is in normal operation. When the MCU does not need to run, an appropriate low-power mode can be selected to save power. The user needs to select an appropriate low-power mode based on conditions such as the lowest power consumption, the fastest startup time and available wake-up events.

The chip provides the following 4 main low-power modes:

- Idle mode

All peripherals remain powered, the core stops running, and the clock system is running. After a wake-up event is detected, it can be woken up immediately.

- Halt mode

On the basis of idle mode, the clock system stops. After a wake-up event is detected, the clock will run first, and then the core will be woken up to run.

- Sleep mode:

The main LDO is turned off, and the ultra-low power ULP-LDO maintains the power supply of PMU, core and basic peripherals. You can select whether to turn on LSE or LSI, and to maintain power supply of RAM2K,

RAM30K, USB and RF configurations. After a wake-up event is detected, first the main LDO is turned on, then the clock will run, and finally the core will be woken up, the program will continue to run, and a higher frequency can be reset when needed.

- Shutdown mode:

Based on the sleep mode, the core and basic peripherals, USB and RF configurations are turned off, and you can select whether to turn on LSE or LSI, and to maintain power supply of RAM2K and RAM30K. After detecting a wake-up event, PMU will perform a GRWSM reset, and the software can distinguish RPOR based on the reset flag RB_RESET_FLAG and the data retained in optional RAM.

The following table describes in detail the characteristics and wake-up means in several low-power modes:

Table 5-2 Low power modes

| Mode | Feature | Entry Conditions | Wake-up Event | Power consumption |
|---|---|---|---|---|
| Idle | The peripherals are powered normally, the core stops running, the clock system is running, but the clocks of each peripheral can be selected to turn off by the peripheral clock control bit. | Set SLEEPDEEP=0, execute WFI() or WFE() after setting the wake-up conditions. | I/O or RTC or BAT or USB | 1.6mA |
| Halt | The peripherals are powered normally, the core stops running, the clock system stops (PLL/HSE stops). | Set SLEEPDEEP=1, execute WFI() or WFE() after setting the wake-up conditions. | I/O or RTC or BAT or USB | 320uA |
| Sleep | The main LDO is off, and the ultra-low power ULP-LDO maintains power supply of PMU, core and basic peripherals, you can select whether to turn on LSE or LSI, and to maintain power supply of RAM2K, RAM30K, USB and RF configurations. | Set SLEEPDEEP=1, execute WFI() or WFE() after setting the wake-up conditions. | I/O or RTC or BAT. The chip will continue to run after woken up. | 0.7uA~2.8uA |
| Shutdown | Ultra-low power consumption LDO maintains power supply of PMU, you can select whether to turn on LSE or LSI, and to maintain power supply of RAM2K and RAM30K for data retention. | Set SLEEPDEEP=1, set POWER_PLAN, execute WFI() or WFE() after setting the wake-up conditions. | I/O or RTC or BAT. The chip will automatically reset after woken up | 0.2uA~2.3uA |

The following table describes the detailed configurations of several low-power modes:

Table 5-3 Detailed configuration example of low-power mode

| Planning configuration | SYS_EN | RAM2K | RAM30K | CK32K | CORE | EXTEND | Power consumption (for reference only) |
|---|---|---|---|---|---|---|---|
| Maintaining | System | Data area | Data area | LSE/LSI | CPU core | USB and RF | PMU and RTC |

| supply function | power VSW | 2KB | 30KB | RTC wake-up | and basic peripherals | configurations | registers are always powered, about 0.2uA |
|---|---|---|---|---|---|---|---|
| Common configurations in shutdown mode | 0 | 0 | 0 | 0 | 0 | 0 | 0.2uA |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0.5uA |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0.5uA |
| | 0 | 1 | 0 | 1 | 0 | 0 | 0.8uA |
| Common configurations in sleep mode | 0 | 1 | 0 | 0 | 1 | 0 | 0.7uA |
| | 0 | 1 | 0 | 1 | 1 | 0 | 1.0uA |
| | 0 | 0 | 1 | 0 | 1 | 0 | 1.8uA |
| | 0 | 0 | 1 | 1 | 1 | 0 | 2.1uA |
| | 0 | 1 | 1 | 1 | 1 | 1 | 2.4uA |

## 5.4 DC-DC operation steps

Enable DC-DC power mode (It is needed to confirm the inductance and capacitance required by DC-DC on the external hardware circuit before enabling)

(1) Enter safe access mode: First write 0x57 to the R8_SAFE_ACCESS_SIG register and then write 0xA8;

(2) Open DC-DC bias circuit: Set the RB_PWR_DCDC_PRE in the R16_POWER_PLAN register to 1;

(3) Turn on the DC-DC power: Set the RB_PWR_DCDC_EN in the R16_POWER_PLAN register to 1, to enable DC-DC.

Disable DC-DC and switch to direct power mode

(1) Enter safe access mode: First write 0x57 to the R8_SAFE_ACCESS_SIG register and then write 0xA8;

(2) Clear the RB_PWR_DCDC_EN and RB_PWR_DCDC_PRE control bits in the R16_POWER_PLAN register.

# Chapter 6 System Clock and RTC

## 6.1 Introduction to system clock

The following different clock sources can be selected to drive the system clock HCLK (Fsys)

— Frequency division of HSE.

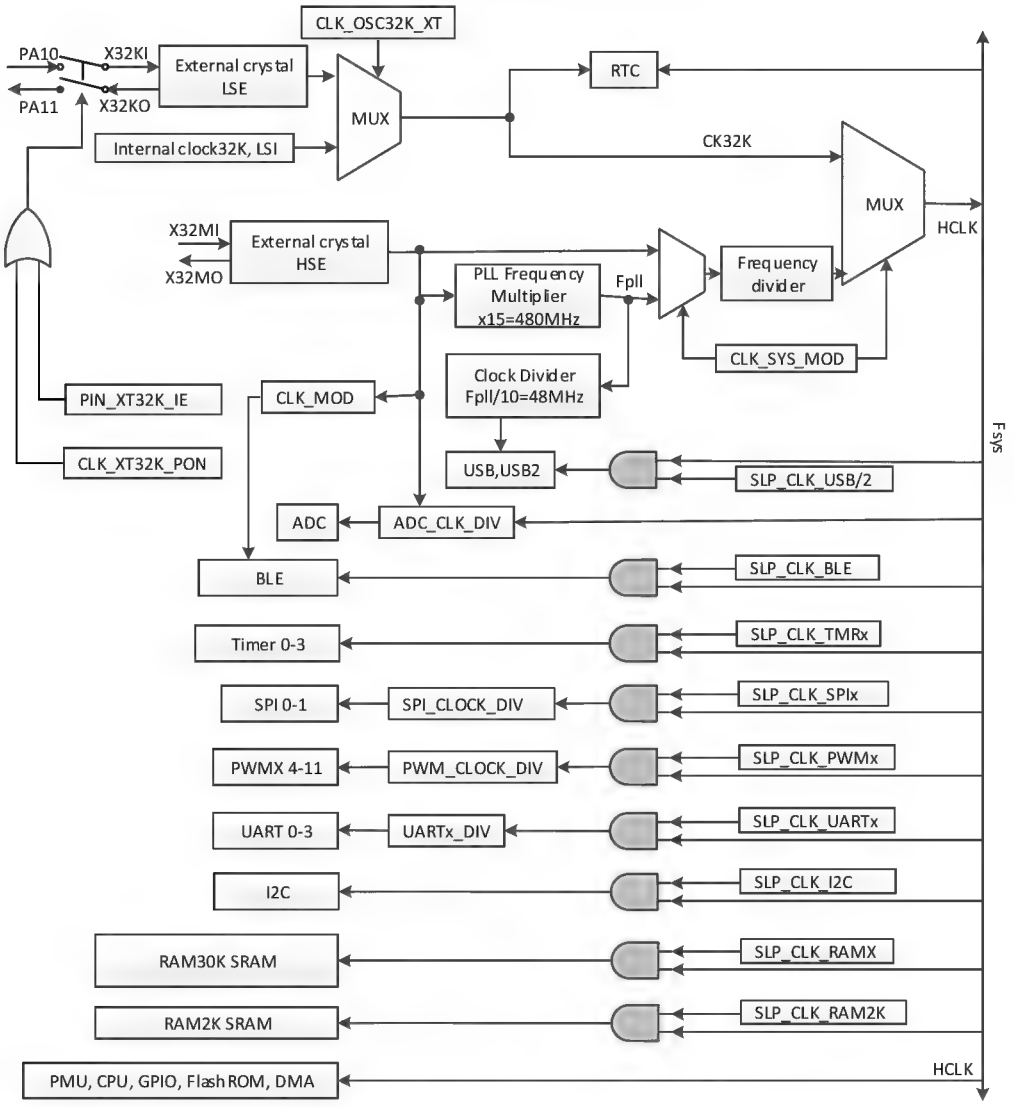
— Internal PLL (480MHz by default) frequency division.

— LSE or LSI original clock CK32K.

Any clock source can be turned on/off independently, thereby optimizing system power consumption.

### 6.1.1 Clock architecture

Figure 6-1 Clock tree block diagram




The figure above is the internal clock tree architecture of the system. The 32KHz clock source (CK32K) is selected for RTC function, so the low-frequency clock must be turned on when using these functions. USB data transfer depends on the clock source generated by PLL. Other peripheral driven clock and digital control logic are driven by the system clock or by frequency re-division.

## 6.2 Introduction to RTC

The Real-time Clock (RTC) is an independent timer that contains a set of counters that can count continuously. Under the corresponding software configuration, a simple calendar function is available. Reset the current time and date by modifying the value of counter.

The RTC register is powered as often as the PMU. After the system is reset or woken up from low power mode, RTC setting and time remain unchanged.

### 6.2.1 Main features

- 2 modes that can be configured:
  — Timing mode: A fixed cycle time (timing) can be selected for the software to generate interrupt notifications.
  — Trigger mode: Match a target alarm clock time preset by the software to generate an interrupt notification.
- Three groups of 16-bit counters that provide count of CK32K primitive cycle, 2s cycle and 1-day cycle.

## 6.3 Register description

Table 6-1 Clock and oscillator control registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_CLK_SYS_CFG | 0x40001008 | System clock configuration register | 0x0005 |
| R8_HFCK_PWR_CTRL | 0x4000100A | High frequency clock module power control register | 0x14 |
| R16_INT32K_TUNE | 0x4000102C | Internal 32KHz clock tune register | 0x1011 |
| R8_XT32K_TUNE | 0x4000102E | External 32KHz clock resonance tune register | 0xC3 |
| R8_CK32K_CONFIG | 0x4000102F | 32KHz oscillator configuration register | 0xX2 |
| R8_XT32M_TUNE | 0x4000104E | External 32MHz clock resonance tune register | 0x32 |
| R16_OSC_CAL_CNT | 0x40001050 | Oscillator frequency calibration count register | 0xXXXX |
| R8_OSC_CAL_OV_CNT | 0x40001052 | Oscillator frequency calibration overflow count register | 0x00 |
| R8_OSC_CAL_CTRL | 0x40001053 | Oscillator frequency calibration control register | 0x09 |
| R8_PLL_CONFIG | 0x4000104B | PLL configuration register | 0x4A |
| R8_RTC_FLAG_CTRL | 0x40001030 | RTC flag and control register | 0x30 |
| R8_RTC_MODE_CTRL | 0x40001031 | RTC mode control register | 0x02 |
| R32_RTC_TRIG | 0x40001034 | RTC trigger value register | 0x00000000 |
| R16_RTC_CNT_32K | 0x40001038 | RTC based 32768Hz count value register | 0xXXXXXXXX |
| R16_RTC_CNT_2S | 0x4000103A | RTC count value register in the unit of 2S | 0xXXXXXXXX |
| R32_RTC_CNT_DAY | 0x4000103C | RTC count value register in the unit of day | 0x0000XXXX |

System clock configuration register (R16_CLK_SYS_CFG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved | 00h |
| [7:6] | RB_CLK_SYS_MOD | RWA | HCLK system clock source mode selection:<br>00/10: CK32M (32MHz by default) for frequency division;<br>01: PLL (480MHz by default) for frequency division;<br>11: CK32K (32KHz by default), used as HCLK. | 00b |
| 5 | Reserved | RO | Reserved | 0 |
| [4:0] | RB_CLK_PLL_DIV | RWA | HCLK output clock frequency division factor, the minimum value is 2. 0 means the maximum value 32. Write 1 to disable HCLK. | 00101b |

Calculation:

Fck32m = XT_32MHz;

Fck32k = RB_CLK_OSC32K_XT ? XT_32KHz : RC_32KHz;

Fpll = Fck32m * 15 = 480MHz;

Fsys = RB_CLK_SYS_MOD==3 ? Fck32k:(RB_CLK_SYS_MOD[0]? Fpll :Fck32m)/RB_CLK_PLL_DIV;

Power-on default value Fsys = Fck32m/ RB_CLK_PLL_DIV = 32MHz / 5 = 6.4MHz;

Fsys range: 32KHz, 2MHz~10MHz, 15MHz~80MHz

High frequency clock module power control register (R8_HFCK_PWR_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:5] | Reserved | RO | Reserved | 000b |
| 4 | RB_CLK_PLL_PON | RWA | PLL power control bit:<br>1: Power on; 0: Power off. | 1 |
| 3 | RB_CLK_XT32M_KEEP | RWA | Used to control clock system to stop in Halt mode:<br>1: In Halt mode, not automatically stop HSE or PLL;<br>0: In Halt mode, automatically stop HSE and PLL. | 0 |
| 2 | RB_CLK_XT32M_PON | RWA | External 32MHz oscillator HSE power control:<br>1: Power on; 0: Power off. | 1 |
| [1:0] | Reserved | RO | Reserved | 00b |

Internal 32KHz clock tune register (R16_INT32K_TUNE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:13] | Reserved | RO | Reserved | 000b |
| [12:0] | RB_INT32K_TUNE | RWA | Internal RC 32KHz clock frequency calibration value. | 1011h |

External 32KHz clock resonance tune register (R8_XT32K_TUNE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:4] | RB_XT32K_C_LOAD | RWA | Select the built-in load capacitor matching the external 32KHz crystal (which may affect the RTC clock accuracy).<br>Capacitance = RB_XT32K_C_LOAD + 12pF.<br>0000b to 1111b correspond to approximately 12pF to 27pF, respectively.<br>Select according to the crystal parameters used. | 1100b |
| [3:2] | Reserved | RO | Reserved | 00b |
| [1:0] | RB_XT32K_I_TUNE | RWA | External 32KHz oscillator bias current select:<br>00: 70% of rated current;<br>01: Rated current;<br>10: 140% of rated current;<br>11: 200% of rated current.<br>The current can be changed to the rated current after the crystal oscillator is stable. | 11b |

32KHz oscillator configuration register (R8_CK32K_CONFIG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_32K_CLK_PIN | RO | 32KHz clock pin status (asynchronous signal). | X |
| [6:4] | Reserved | RO | Reserved | 000b |
| 3 | RB_CLK_OSC32K_FILT | RWA | Internal 32KHz oscillator noise filter mode:<br>1: Disable; 0: Enable. | 0 |
| 2 | RB_CLK_OSC32K_XT | RWA | CK32K（32KHz）clock source select bit:<br>1: External 32KHz oscillator;<br>0: Internal 32KHz oscillator. | 0 |
| 1 | RB_CLK_INT32K_PON | RWA | Internal 32KHz oscillator power control bit:<br>1: Power on; 0: Power off. | 1 |
| 0 | RB_CLK_XT32K_PON | RWA | External 32KHz oscillator power control bit:<br>1: Power on; 0: Power off. | 0 |

External 32MHz clock resonance tune register (R8_XT32M_TUNE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| [6:4] | RB_XT32M_C_LOAD | RWA | Select the built-in load capacitor that matches the external 32MHz crystal<br>(Which may affect wireless communication):<br>Capacity=RB_XT32M_C_LOAD*2+10pF,<br>000b~111b correspond to approximately 10pF~24pF respectively.<br>Select according to the parameters of crystal used; the common value is 111b. | 011b |

| | | | | |
|---|---|---|---|---|
| [3:2] | Reserved | RO | Reserved | 00b |
| [1:0] | RB_XT32M_I_BIAS | RWA | External 32MHz oscillator bias current select:<br>00: 75% of rated current;<br>01: Rated current;<br>10: 125% of rated current;<br>11: 150% of rated current. | 10b |

Oscillator frequency calibration count register (R16_OSC_CAL_CNT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | RB_OSC_CAL_IF | RW1 | Oscillator capture complete interrupt flag bit, write 1 to clear it:<br>1: Interrupt; 0: No interrupt. | 0 |
| 14 | RB_OSC_CAL_OV_CLR | RW1 | R8_OSC_CAL_OV_CNT register value non-zero indicator, write 1 to clear R8_OSC_CAL_OV_CNT. | 0 |
| [13:0] | RB_OSC_CAL_CNT | RO | Count value based on system clock frequency of multiple CK32K cycles, used to calibrate internal 32KHz oscillator frequency. | XXXXh |

Oscillator frequency calibration overflow count register (R8_OSC_CAL_OV_CNT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | RB_OSC_CAL_OV_CNT | RO | Oscillator frequency calibration count overflow times, write 1 to RB_OSC_CAL_OV_CLR to clear this register. | XXh |

Oscillator frequency calibration control register (R8_OSC_CAL_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| 6 | RB_OSC_CNT_END | RWA | Oscillator capture end-point selection:<br>1: 2 additional cycles; 0: No. | 0 |
| 5 | RB_OSC_CNT_EN | RWA | Oscillator frequency calibration counter enable:<br>1: Enable counting; 0: Disable counting. | 0 |
| 4 | RB_OSC_CAL_IE | RWA | Oscillator capture complete interrupt enable:<br>1: Enable; 0: Disable. | 0 |
| 3 | RB_OSC_CNT_HALT | RO | Oscillator frequency calibration counter count status:<br>1: Counting is being paused;<br>0: Counting is in progress. | 1 |
| [2:0] | RB_OSC_CNT_TOTAL | RWA | Oscillator capture total cycle selection:<br>000: 1 cycle; 001: 2 cycles;<br>010: 4 cycles; 011: 32 cycles;<br>100: 64 cycles; 101： 128 cycles; | 001b |

| | | | | |
|---|---|---|---|---|
| | | | 110: 1024 cycles; 111: 2047 cycles. | |

PLL configuration register (R8_PLL_CONFIG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_FLASH_IO_MOD | RWA | FlashROM control operation mode | 0 |
| [6:0] | RB_PLL_CFG_DAT | RWA | PLL configuration parameters. | 1001010b |

RTC flag and control register (R8_RTC_FLAG_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_RTC_TRIG_FLAG | RO | RTC trigger mode activation flag. | 0 |
| 6 | RB_RTC_TMR_FLAG | RO | RTC timing mode activation flag. | 0 |
| 5 | RB_RTC_TRIG_CLR | RW | This bit is always 1 when the trigger mode is disabled.<br>When the trigger mode is enabled, write 1, clear the trigger mode activation flag RB_RTC_TRIG_FLAG and automatically cleared to 0. | 1 |
| 4 | RB_RTC_TMR_CLR | RW | When the timing mode is disabled, this bit is fixed as 1.<br>When the timing mode is enabled, write 1, clear the timing mode activation flag RB_RTC_TMR_FLAG and automatically cleared to 0. | 1 |
| [3:0] | Reserved | RO | Reserved | 0000b |

RTC mode control register (R8_RTC_MODE_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_RTC_LOAD_HI | RWA | Write 1 to load the high word of RTC counter, and automatically cleared to 0 after loading. Load R32_RTC_TRIG (actually only the low 14 bits) to R32_RTC_CNT_DAY. | 0 |
| 6 | RB_RTC_LOAD_LO | RWA | Write 1 to load the low word of RTC counter, and automatically cleared after loading. Load the high 16 bits of R32_RTC_TRIG to R16_RTC_CNT_2S; load the low 16 bits of R32_RTC_TRIG to R16_RTC_CNT_32K. | 0 |
| 5 | RB_RTC_TRIG_EN | RWA | RTC trigger mode enable:<br>1: Enable; 0: Disable. | 0 |
| 4 | RB_RTC_TMR_EN | RWA | RTC timing mode enable:<br>1: Enable; 0: Disable. | 0 |
| 3 | RB_RTC_IGNORE_B0 | RWA | Ignore and compare the lowest bit of matching value in trigger mode:<br>1: Ignore the lowest bit;<br>0: Compare the lowest bit. | 0 |

| | | | | |
|---|---|---|---|---|
| [2:0] | RB_RTC_TMR_MODE | RWA | RTC timing mode fixed cycle (timing) selection:<br>000: 0.125S; 001: 0.25S;<br>010: 0.5S; 011: 1S;<br>100: 2S; 101: 4S;<br>110: 8S; 111: 16S. | 010b |

RTC trigger value register (R32_RTC_TRIG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | R32_RTC_TRIG | RWA | The preset matching value in RTC trigger mode, and the high 16 bits and low 16 bits are matched with R16_RTC_CNT_2S and R16_RTC_CNT_32K respectively.<br>Cooperate with RB_RTC_LOAD_LO and RB_RTC_LOAD_HI to update the current value of RTC counter. | 0000h |

*Note: The preset matching value is not directly written into the target time, and it involves simple calculations. Please refer to the following instructions.*

RTC count value register based on 32768Hz (R16_RTC_CNT_32K)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_RTC_CNT_32K | RO | RTC count value register based on 32768Hz. | XXXXh |

RTC count value register in the unit of 2S (R16_RTC_CNT_2S)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_RTC_CNT_2S | RO | The current count value of the RTC in 2S units. | XXXXh |

RTC count value register in the unit of day (R32_RTC_CNT_DAY)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:14] | Reserved | RO | Reserved | 00b |
| [13:0] | R32_RTC_CNT_DAY | RO | RTC current count value in the unit of day. | XXXXXXX<br>XXXXXXXb |

## 6.4 Functional description and configuration

### 6.4.1 RTC counter initialization

(1) Set the value of the R32_RTC_TRIG register and set RB_RTC_LOAD_HI, to load the value of the R32_RTC_TRIG register into the R32_RTC_CNT_DAY register;

(2) Set the value of the R32_RTC_TRIG register and set RB_RTC_LOAD_LO, to load the value of the high 16 bits and the low 16 bits of the R32_RTC_TRIG register into the R16_RTC_CNT_2S register and the R16_RTC_CNT_32K register respectively.

### 6.4.2 Switch RTC clock source to LSE crystal

(1) Confirm that the GPIO pins where X32KI and X32KO are located are not set as outputs, with no pull-up and pull-down resistors, and only crystals are provided;
(2) Configure the R8_CK32K_CONFIG register, set RB_CLK_XT32K_PON to 1, to enable the external 32KHz crystal oscillator;
(3) It is recommended to set RB_XT32K_I_TUNE to the maximum first, and wait for the crystal oscillator to stabilize (about several hundreds of mS) and then change to the rated current;
(4) Configure the R8_CK32K_CONFIG register, set RB_CLK_OSC32K_XT to 1, and switch clock source to the crystal oscillator;
(5) Wait for at least half of the 32KHz clock cycle, usually 16uS, to actually finish the switch of clock source.

### 6.4.3 RTC timing function

(1) Configure R8_RTC_MODE_CTRL register, set RB_RTC_TMR_MODE to select the appropriate timing period, set RB_RTC_TMR_EN to 1, and turn on RTC timing function;
(2) After reaching the timing period, RTC timing activation flag RB_RTC_TMR_FLAG and interrupt will be generated; check R8_RTC_FLAG_CTRL register and set RB_RTC_TMR_CLR to clear the flag.

### 6.4.4 RTC trigger function

(1) Set the target matching value in R32_RTC_TRIG register, and see the calculation and operation steps: Calculate the target time value by taking the current time R32_RTC_CNT_32K (high 16 bits R16_RTC_CNT_2S and low 16 bits R16_RTC_CNT_32K) plus the interval time DelayTime (in the unit of S), T32 = R32_RTC_CNT_32K + DelayTime * 32768,
Write T32 into the R32_RTC_TRIG register to complete the matching value setting;
(2) Configure R8_RTC_MODE_CTRL, set RB_RTC_TRIG_EN to 1, and turn on RTC trigger function;
(3) When the current RTC count values R16_RTC_CNT_2S and R16_RTC_CNT_32K respectively match the preset high and low 16 bits of R32_RTC_TRIG, RTC trigger activation flag RB_RTC_TRIG_FLAG and interrupt are generated, and the flag can be cleared by setting RB_RTC_TRIG_CLR.
(4) If the RTC time has been calibrated, target absolute time trigger is supported, and the target time value T32 is calculated based on the target year/month/day/hour/minute/second/millisecond. Other steps are the same as above. Please refer to the evaluation board example program for details.

### 6.4.5 Calibrate internal 32K clock LSI with HSE

Refer to evaluation board example program.

# Chapter 7 General-purpose I/O and Alternate Functions

## 7.1 Introduction to GPIO

The chip provides 2 sets of GPIO ports, PA and PB, with a total of 40 GPIO pins. Among 40 GPIO pins, 32 GPIO pins have interrupt and wake-up functions, and parts of pins have alternate and mapping functions.

Each GPIO port has a 32-bit direction configuration register (R32_Px_DIR), a 32-bit pin input register (R32_Px_PIN), a 32-bit data output register (R32_Px_OUT), a 32-bit data reset register (R32_Px_CLR), a 32-bit pull-up resistor configuration register (R32_Px_PU), a 32-bit pull-down resistor/drive capability configuration register (R32_Px_PD_DRV).

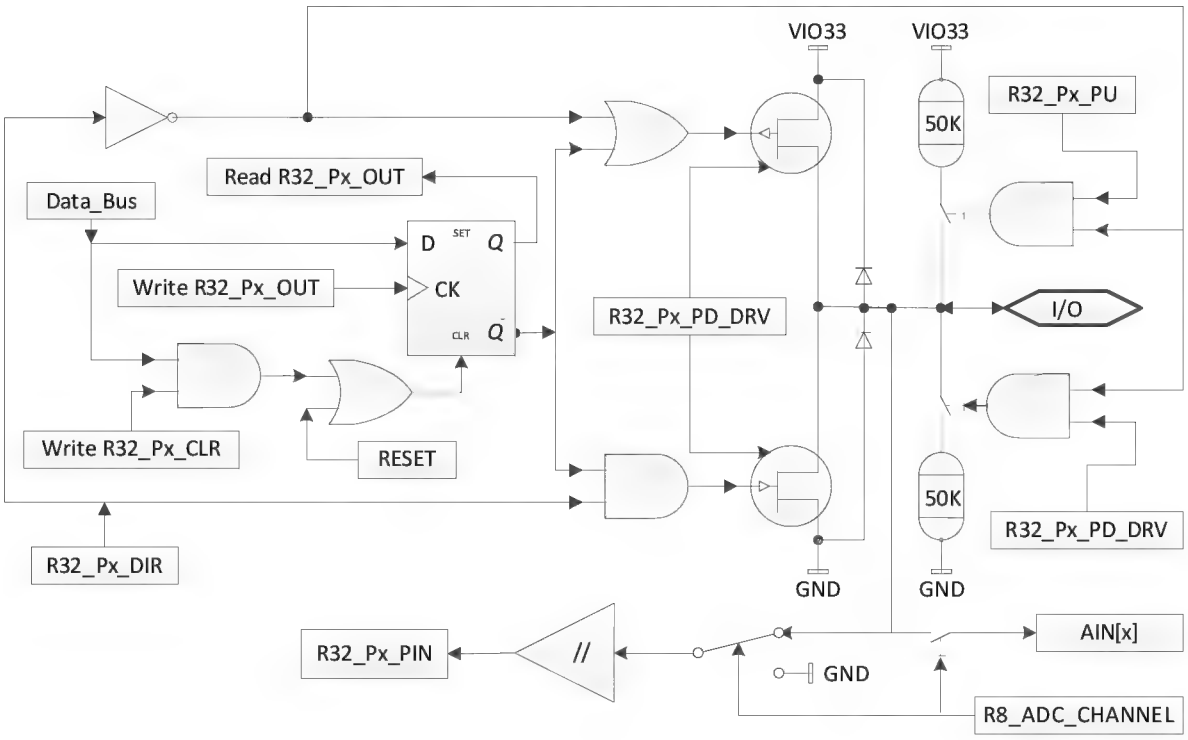
In PA port, PA[0]～PA[15] bits are valid, corresponding to 16 GPIO pins on the chip.

In PB port, PB[0]～PB[23] bits are valid, corresponding to 24 GPIO pins on the chip.

Each I/O port bit can be freely programmed, but the I/O port register must be accessed by 8-bit, 16-bit or 32-bit words. If the alternate function of pin is not enabled, it will be used as a general-purpose I/O port by default.

The following figure is the block diagram of GPIO internal architecture:

Figure 7-1 I/O internal architecture block diagram




## 7.2 External interrupt/wakeup

Parts of I/O pins of the chip have interrupt function and can realize sleep and wakeup.

In order to use external interrupts, the port bits must be configured in input mode. And provide 4 kinds of trigger modes: high level, low level, rising edge, falling edge.

The wake-up function needs to enable the interrupt R16_Px_INT_EN of the port bit, and turn on the GPIO wake-up control bit RB_SLP_GPIO_WAKE in the R8_SLP_WAKE_CTRL register.

## 7.3 GPIO alternate and remapping

### 7.3.1 Alternate functions

Some I/O pins have alternate functions. After power on, all I/O pins are as GPIO by default. After enabling various functional modules, the corresponding original GPIO pins are configured as corresponding functional pins of each functional module.

If a pin has multiple alternate functions, and multiple functions are enabled, please refer to the function order in the "Alternate Function" list in the pin description in section 1.2 for the priority order of alternate functions.

For example: If the PB23 pin is alternate as RST#/TMR0_/TXD2/PWM11, RST# reset input function has a high priority, and PWM11 output function has the lowest priority. In this way, the alternate functions with the relatively higher priority of the pin whose functions with the lowest priority need not to be used can be enabled among multiple alternate functions.

The following tables list some GPIO configurations of some functional pins which are used for peripheral modules.

Table 7-1 Timer x

| TMR0/1/2/3 pin | Functional configuration | GPIO configuration |
|---|---|---|
| TMRx | Input capture channel x | Input (floating input/pull-up input/pull-down input) |
| | Output PWM channel x | Push-pull output |

Table 7-2 UARTx

| UART0/1/2/3 pin | Functional configuration | GPIO configuration |
|---|---|---|
| TXDx | UART transmit x | Push-pull output |
| RXDx | UART receive x | Pull-up input (recommended) or floating input |
| RTS，DTR | MODEM signal output or RS485 control | Push-pull output |
| CTS，DSR，RI，DCD | MODEM signal input | Pull-up input (recommended) or floating input |

Table 7-3 SPIx

| SPI0/1 pin | Functional configuration | GPIO configuration |
|---|---|---|
| SCKx | Clock output in master mode | Push-pull output |
| | Clock input in slave mode | Input (floating input/pull-up input/pull-down input) |
| MOSIx | Full-duplex mode-master mode | Push-pull output |
| | Full-duplex mode-slave mode | Input (floating input/pull-up input/pull-down input) |

| | | |
|---|---|---|
| | Half-duplex mode-master mode | Not used, can be used as general purpose I/O |
| | Half-duplex mode-slave mode | Not used, can be used as general purpose I/O |
| MISOx | Full-duplex mode-master mode | Input (floating input/pull-up input/pull-down input) |
| | Full duplex mode-slave mode | Input (pull-up is recommended, automatically switched to push-pull output after chip select) or push-pull output (it is forbidden to be used for bus connection) |
| | Half duplex mode-master mode | Input or push-pull output, manual switching |
| | Half duplex mode-slave mode | Input (pull-up is recommended, automatically switched to push-pull output after chip select) |
| SCS | Chip select output in master mode | Push-pull output (can be replaced with other pins) |
| | Chip select input in slave mode | Pull-up input (recommended) or floating input |

Table 7-4 I2C

| I2C pin | Functional configuration | GPIO configuration |
|---|---|---|
| SCL | Serial clock output – master mode | Push-pull output (multi-master is not supported in this mode) |
| | Serial clock output/input – multi-master mode | Input (pull-up is recommended, automatically open-drain output when needed) |
| | Serial clock input – slave mode | Pull-up input (recommended) or floating input |
| SDA | Serial data input/output | Input (pull-up is recommended, automatically open-drain output when needed) |

Table 7-5 ADC

| ADC sample channel pin | Functional configuration | GPIO configuration |
|---|---|---|
| AINx | ADC input channel | Floating input |

Table 7-6 USBx

| USB signal pin | Functional configuration | GPIO configuration |
|---|---|---|
| UD-, U2D- | Connected to internal USB transceiver | Floating input |
| UD+, U2D+ | Connected to internal USB transceiver | Floating input |

### 7.3.2 Remapping of function pins

In order to enable the peripheral functions and optimize the utilization rate at the same time, some function pins can be remapped to other pins by setting the function pin remapping register (R16_PIN_ALTERNATE).

Table 7-7 Alternate function remapping pins

| Peripheral function pins | Default GPIO pins | Remapped GPIO pins |
|---|---|---|
| SCS/SCK0/MOSI/MISO | PA[12]/PA[13]/PA[14]/PA[15] | PB[12]/PB[13]/PB[14]/PB[15] |

| | | |
|---|---|---|
| RXD3/TXD3 | PA[4]/PA[5] | PB[20]/PB[21] |
| RXD2/TXD2 | PA[6]/PA[7] | PB[22]/PB[23] |
| RXD1/TXD1 | PA[8]/PA[9] | PB[12]/PB[13] |
| RXD0/TXD0/DSR/DTR | PB[4]/PB[7]/PB[1]/PB[5] | PA[15]/PA[14]/PB[14]/PB[15] |
| TMR3/PWM3/CAP3 | PB[22] | PA[2] |
| TMR2/PWM2/CAP2 | PA[11] | PB[11] |
| TMR1/PWM1/CAP1 | PA[10] | PB[10] |
| TMR0/PWM0/CAP0 | PA[9] | PB[23] |
| PWM4/PWM5 | PA[12]/PA[13] | PA[6]/PA[7] |
| PWM7/PWM8/PWM9 | PB[4]/PB[6]/PB[7] | PB[1]/PB[2]/PB[3] |
| SCL/SDA | PB[13]/PB[12] | PB[21]/PB[20] |

## 7.4 Register description

Table 7-8 GPIO registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_PIN_ALTERNATE | 0x40001018 | Functional pin remapping register | 0x0000 |
| R16_PIN_ANALOG_IE | 0x4000101A | Peripheral analog pin configuration register | 0x0000 |
| R16_PA_INT_EN | 0x40001090 | PA port interrupt enable register | 0x0000 |
| R16_PB_INT_EN | 0x40001092 | PB port interrupt enable register | 0x0000 |
| R16_PA_INT_MODE | 0x40001094 | PA port interrupt mode configuration register | 0x0000 |
| R16_PB_INT_MODE | 0x40001096 | PB port interrupt mode configuration register | 0x0000 |
| R16_PA_INT_IF | 0x4000109C | PA port interrupt flag register | 0x0000 |
| R16_PB_INT_IF | 0x4000109E | PB port interrupt flag register | 0x0000 |
| R32_PA_DIR | 0x400010A0 | PA port direction configuration register | 0x00000000 |
| R32_PA_PIN | 0x400010A4 | PA port pin input register | 0x0000XXXX |
| R32_PA_OUT | 0x400010A8 | PA port data output register | 0x00000000 |
| R32_PA_CLR | 0x400010AC | PA port data reset register | 0x00000000 |
| R32_PA_PU | 0x400010B0 | PA port pull-up resistor configuration register | 0x00000000 |
| R32_PA_PD_DRV | 0x400010B4 | PA port pull-down/drive configuration register | 0x00000000 |
| R32_PB_DIR | 0x400010C0 | PB port direction configuration register | 0x00000000 |
| R32_PB_PIN | 0x400010C4 | PB port pin input register | 0x00XXXXXX |
| R32_PB_OUT | 0x400010C8 | PB port data output register | 0x00000000 |
| R32_PB_CLR | 0x400010CC | PB port data reset register | 0x00000000 |
| R32_PB_PU | 0x400010D0 | PB port pull-up resistor configuration register | 0x00000000 |
| R32_PB_PD_DRV | 0x400010D4 | PB port pull-down/drive configuration register | 0x00000000 |

Functional pin remapping register (R16_PIN_ALTERNATE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | RB_RF_ANT_SW_EN | RW | RF antenna switch control output enable:<br>1: Switch control output to PB[16]~ PB[21];<br>0: Disable output. | 0 |
| 14 | RB_PIN_U0_INV | RW | UART0 input/output inverted enable: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: RXD0/RXD0_ inverted input, TXD0/TXD0_ inverted output;<br>0: Normal non-inverted input/output. | |
| 13 | RB_PIN_INTX | RW | INT24/INT25 functional pin mapping select bit:<br>1: INT24_/25_ is mapped to PB[22]/PB[23];<br>0: INT24/25 is mapped to PB[8]/PB[9].<br>*Note: INT24/INT25 is corresponding interrupt input of [9:8] in R16_PB_INT_EN, R16_PB_INT_MODE, R16_PB_INT_IF.* | 0 |
| 12 | RB_PIN_MODEM | RW | UART0 MODEM functional pin mapping select bit:<br>1: DSR_/DTR_ is mapped to PB[14]/PB[15];<br>0: DSR/DTR is mapped to PB[1]/PB[5]. | 0 |
| 11 | RB_PIN_I2C | RW | I2C functional pin mapping select bit:<br>1: SCL_/SDA_ is mapped to PB[21]/PB[20];<br>0: SCL/SDA is mapped to PB[13]/PB[12]. | 0 |
| 10 | RB_PIN_PWMX | RW | PWMx functional pin mapping select bit:<br>1: PWM4/5/7/8/9 is mapped to PA[6]/PA[7]/PB[1]/PB[2]/PB[3];<br>0: PWM4/5/7/8/9 is mapped to PA[12]/PA[13]/PB[4]/PB[6]/PB[7]. | 0 |
| 9 | Reserved | RO | Reserved | 0 |
| 8 | RB_PIN_SPI0 | RW | SPI0 functional pin mapping select bit:<br>1: SCK0_/SCS_/MOSI_/MISO_ is mapped to PB[12]/PB[13]/PB[14]/PB[15];<br>0: SCK0/SCS/MOSI/MISO is mapped to PA[12]/PA[13]/PA[14]/PA[15]. | 0 |
| 7 | RB_PIN_UART3 | RW | UART3 functional pin mapping select bit:<br>1: RXD3_/TXD3_ is mapped to PB[20]/PB[21];<br>0: RXD3/TXD3 is mapped to PA[4]/PA[5]. | 0 |
| 6 | RB_PIN_UART2 | RW | UART2 functional pin mapping select bit:<br>1: RXD2_/TXD2_ is mapped to PB[22]/PB[23];<br>0: RXD2/TXD2 is mapped to PA[6]/PA[7]. | 0 |
| 5 | RB_PIN_UART1 | RW | UART1 functional pin mapping select bit:<br>1: RXD1_/TXD1_ is mapped to PB[12]/PB[13];<br>0: RXD1/TXD1 is mapped to PA[8]/PA[9]. | 0 |
| 4 | RB_PIN_UART0 | RW | UART0 functional pin mapping select bit:<br>1: RXD0_/TXD0_ is mapped to PA[15]/PA[14];<br>0: RXD0/TXD0 is mapped to PB[4]/PB[7]. | 0 |
| 3 | RB_PIN_TMR3 | RW | TMR3 functional pin mapping select bit:<br>1: TMR3_/PWM3_/CAP3_ is mapped to PA[2];<br>0: TMR3/PWM3/CAP3 is mapped to PB[22]. | 0 |
| 2 | RB_PIN_TMR2 | RW | TMR2 functional pin mapping select bit:<br>1: TMR2_/PWM2_/CAP2_ is mapped to PB[11]; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: TMR2/PWM2/CAP2 is mapped to PA[11]. | |
| 1 | RB_PIN_TMR1 | RW | TMR1 functional pin mapping select bit:<br>1：TMR1_/PWM1_/CAP1_ is mapped to PB[10]；<br>0： TMR1/PWM1/CAP1 is mapped to PA[10]。 | 0 |
| 0 | RB_PIN_TMR0 | RW | TMR0 functional pin mapping select bit:<br>1: TMR0_/PWM0_/CAP0_ is mapped to PB[23];<br>0: TMR0/PWM0/CAP0 is mapped to PA[9]. | 0 |

Peripheral analog pin configuration register (R16_PIN_ANALOG_IE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | RB_PIN_ADC4_5_IE | RW | ADC/TKEY 4/5 channel pin digital input disable:<br>1: Disable PA14-15 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 14 | RB_PIN_ADC2_3_IE | RW | ADC/TKEY 2/3 channel pin digital input disable:<br>1: Disable PA12-13 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 13 | RB_PIN_XT32K_IE | RW | 32KHz crystal LSE pin digital input disable:<br>1: Disable PA10-11 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 12 | RB_PIN_ADC13_IE | RW | ADC/TKEY 13 channel pin digital input disable:<br>1: Disable PA9 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 11 | RB_PIN_ADC12_IE | RW | ADC/TKEY 12 channel pin digital input disable:<br>1: Disable the PA8 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 10 | RB_PIN_ADC1_IE | RW | ADC/TKEY 1 channel pin digital input disable:<br>1: Disable PA5 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 9 | RB_PIN_ADC0_IE | RW | ADC/TKEY 0 channel pin digital input disable:<br>1: Disable PA4 digital input to save power consumption;<br>0: Enable digital input. | 0 |
| 8 | Reserved | RW | Reserved | 0 |
| 7 | RB_PIN_USB_IE | RW | USB pin enable:<br>1: PB10 and PB11 are USB communication pins;<br>0: PB10 and PB11 are not used for USB communication. | 0 |
| 6 | RB_PIN_USB_DP_PU | RW | USB UD+ pin internal pull-up resistor enable: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Forced to enable pull-up (RB_UC_DEV_PU_EN does not work in sleep or power-down mode, so replace it);<br>0: RB_UC_DEV_PU_EN controls pull-up. | |
| 5 | RB_PIN_USB2_IE | RW | USB2 pin enable:<br>1: PB12 and PB13 are USB2 communication pins;<br>0: PB12 and PB13 are not used for USB2 communication. | 0 |
| 4 | RB_PIN_USB2_DP_PU | RW | USB2 U2D+ pin internal pull-up resistor enable:<br>1: Forced to enable pull-up (RB_UC_DEV_PU_EN does not work in sleep or power-down mode, so replace it);<br>0: RB_UC_DEV_PU_EN controls pull-up. | 0 |
| 3 | RB_PIN_ADC11_IE | RW | ADC/TKEY 11 channel pin digital input disable:<br>1: Disable PA7 digital input, to save power consumption;<br>0: Enable digital input. | 0 |
| 2 | RB_PIN_ADC10_IE | RW | ADC/TKEY 10 channel pin digital input disable:<br>1: Disable PA6 digital input, to save power consumption;<br>0: Enable digital input. | 0 |
| 1 | RB_PIN_ADC6_7_IE | RW | ADC/TKEY 7/6 channel pin digital input disable:<br>1: Disable PA2/PA3 digital input, to save power consumption;<br>0: Enable digital input. | 0 |
| 0 | RB_PIN_ADC8_9_IE | RW | ADC/TKEY 9/8 channel pin digital input disable:<br>1: Disable PA0/PA1 digital input, to save power consumption;<br>0: Enable digital input. | 0 |

*Note: If the pin is used for analog function (ADC/TouchKey), it is recommended to turn off the digital input function of the pin, namely to set the digital input as disabled, thus reducing power consumption and helping reduce interference.*

PA port interrupt enable register (R16_PA_INT_EN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_PA_INT_EN | RW | PA pin interrupt enable bit:<br>1: Enable the corresponding interrupt;<br>0: Disable the corresponding interrupt. | 0000h |

PB port interrupt enable register (R16_PB_INT_EN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_PB_INT_EN | RW | PB pin interrupt enable bit:<br>1: Enable the corresponding interrupt; | 0000h |

| | | | | |
|---|---|---|---|---|
| | | | 0: Disable the corresponding interrupt. | |

*Note: R16_PB_INT_EN[9:8] are determined by RB_PIN_INTX selection to correspond to PB[23:22] or PB[9:8].*

PA port interrupt mode configuration register (R16_PA_INT_MODE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_PA_INT_MODE | RW | PA pin interrupt mode select bit:<br>1: Edge trigger; 0: Level trigger. | 0000h |

PB port interrupt mode configuration register (R16_PB_INT_MODE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_PB_INT_MODE | RW | PB pin interrupt mode select bit:<br>1: Edge trigger; 0: Level trigger. | 0000h |

*Note: R16_PB_INT_MODE[9:8] are determined by RB_PIN_INTX selection to correspond to PB[23:22] or PB[9:8].*

PA port interrupt flag register (R16_PA_INT_IF)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_PA_INT_IF | RW1 | PA pin interrupt flag bit, write 1 to clear:<br>1: Interrupt; 0: No interrupt. | 0000h |

PB port interrupt flag register (R16_PB_INT_IF)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_PB_INT_IF | RW1 | PB pin interrupt flag bit, write 1 to clear:<br>1: Interrupt; 0: No interrupt. | 0000h |

*Note: R16_PB_INT_IF[9:8] are determined by RB_PIN_INTX selection to correspond to PB[23:22] or PB[9:8].*

PA port direction configuration register (R32_PA_DIR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:8] | R8_PA_DIR_1 | RW | Current input/output direction configure of PA pin:<br>1: The pin is in output mode;<br>0: The pin is in input mode. | 00h |
| [7:0] | R8_PA_DIR_0 | RW | | 00h |

PA port pin input register (R32_PA_PIN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:8] | R8_PA_PIN_1 | RO | Current level status of PA pin (valid only when R32_PA_DIR corresponding bit is 0):<br>1: Pin input is at high level;<br>0: Pin input is at low level. | XXh |
| [7:0] | R8_PA_PIN_0 | RO | | XXh |

PA port data output register (R32_PA_OUT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:8] | R8_PA_OUT_1 | RW | When the corresponding bit of direction register R32_PA_DIR is 1:<br>Control PA pin output level status:<br>1: Output high level;<br>0: Output low level.<br>When the corresponding bit of direction register R32_PA_DIR is 0:<br>Control PA pin interrupt polarity select:<br>1: High level/rising edge;<br>0: Low level/falling edge. | 00h |
| [7:0] | R8_PA_OUT_0 | RW | | 00h |

PA port data reset register (R32_PA_CLR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:8] | R8_PA_CLR_1 | WZ | PA data register reset control:<br>1: The corresponding bit data of R32_PA_OUT is cleared to 0;<br>0: No effect. | 00h |
| [7:0] | R8_PA_CLR_0 | WZ | | 00h |

PA port pull-up resistor configuration register (R32_PA_PU)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:8] | R8_PA_PU_1 | RW | PA pin pull-up resistor enable control:<br>1: Enable the pull-up resistor;<br>0: Disable the pull-up resistor. | 00h |
| [7:0] | R8_PA_PU_0 | RW | | 00h |

PA port pull-down/drive configuration register (R32_PA_PD_DRV)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:16] | Reserved | RO | Reserved | 0000h |
| [15:8] | R8_PA_PD_DRV_1 | RW | When the corresponding bit of direction register R32_PA_DIR is 0:<br>PA pin pull-down resistor enable control:<br>1: Enable the pull-down resistor;<br>0: Disable the pull-down resistor.<br>When the corresponding bit of direction register R32_PA_DIR is 1:<br>PA pin current drive capability select:<br>1: 20mA level;<br>0: 5mA level. | 00h |
| [7:0] | R8_PA_PD_DRV_0 | RW | | 00h |

PB port direction configuration register (R32_PB_DIR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved | 00h |
| [23:16] | R8_PB_DIR_2 | RW | Current input/output direction configure of PB pin:<br>1: The pin is in output mode;<br>0: The pin is in input mode. | 00h |
| [15:8] | R8_PB_DIR_1 | RW | | 00h |
| [7:0] | R8_PB_DIR_0 | RW | | 00h |

PB port pin input register (R32_PB_PIN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved | 00h |
| [23:16] | R8_PB_PIN_2 | RO | Current level state of PB pin (only when the corresponding bit of R32_PB_DIR is 0, the bit value is valid):<br>1: Pin input is at high level;<br>0: Pin input is at low level. | XXh |
| [15:8] | R8_PB_PIN_1 | RO | | XXh |
| [7:0] | R8_PB_PIN_0 | RO | | XXh |

PB port data output register (R32_PB_OUT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved | 00h |
| [23:16] | R8_PB_OUT_2 | RW | When the corresponding bit of direction register R32_PB_DIR is 1:<br>Control PB pin output level status:<br>1: Output high level;<br>0: Output low level. | 00h |
| [15:8] | R8_PB_OUT_1 | RW | When the corresponding bit of direction register R32_PB_DIR is 1:<br>Control PB pin output level status:<br>1: Output high level;<br>0: Output low level.<br>When the corresponding bit of direction register R32_PB_DIR is 0:<br>Control PB pin interrupt polarity select:<br>1: High level/rising edge;<br>0: Low level/falling edge. | 00h |
| [7:0] | R8_PB_OUT_0 | RW | | 00h |

PB port data reset register (R32_PB_CLR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved | 00h |
| [23:16] | R8_PB_CLR_2 | WZ | PB data register reset control:<br>1: The corresponding bit data of R32_PB_OUT is cleared to 0;<br>0: No effect. | 00h |
| [15:8] | R8_PB_CLR_1 | WZ | | 00h |
| [7:0] | R8_PB_CLR_0 | WZ | | 00h |

PB port pull-up resistor configuration register (R32_PB_PU)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved | 00h |
| [23:16] | R8_PB_PU_2 | RW | PB pin pull-up resistor enable control:<br>1: Enable the pull-up resistor;<br>0: Disable the pull-up resistor. | 00h |
| [15:8] | R8_PB_PU_1 | RW | | 00h |
| [7:0] | R8_PB_PU_0 | RW | | 00h |

PB port pull-down/drive configuration register (R32_PB_PD_DRV)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | Reserved | RO | Reserved | 00h |
| [23:16] | R8_PB_PD_DRV_2 | RW | When the corresponding bit of direction register R32_PB_DIR is 0:<br>PB pin pull-down resistor enable control:<br>1: Enable the pull-down resistor;<br>0: Disable the pull-down resistor.<br>When the corresponding bit of direction register R32_PB_DIR is 1:<br>PB pin current drive capability select:<br>1: 20mA level;<br>0: 5mA level. | 00h |
| [15:8] | R8_PB_PD_DRV_1 | RW | | 00h |
| [7:0] | R8_PB_PD_DRV_0 | RW | | 00h |

## 7.5 Mode configuration of GPIO pins

Each GPIO can be configured to 5 modes, as shown in the table below:

Table 7-9 Port configuration

| Mode | R32_Px_DIR | R32_Px_PU | R32_Px_PD_DRV |
|---|---|---|---|
| Floating input/high impedance input/analog input | 0 | 0 | 0 |
| Input with pull-up resistor | 0 | 1 | 0 |
| Input with pull-down resistor | 0 | 0 | 1 |
| Push-pull output, 5mA level drive capability: | 1 | X | 0 |
| Push-pull output, 20mA level drive capability: | 1 | X | 1 |

# Chapter 8 General-purpose Timer (TMRx)

## 8.1 Introduction to TMRx

The chip is equipped with four 26-bit timers, TMR0, TMR1, TMR2 and TMR3, and the longest timing interval is 2^26 clock cycles. It is applicable to various occasions, including measuring the length of input signal pulse (input capture) or generating output waveform (PWM). In addition, TMR1 and TMR2 support DMA function. Each timer is completely independent and can be operated simultaneously.

### 8.1.1 Alternate functions

- 4 × 26-bit timers, and the longest timing interval is 2^26 clock cycles.
- Timer interrupt is supported, and among them TMR1 and TMR2 support DMA and interrupt.
- Support capture function to measure input pulse length or cycle.
- The capture function can be set to be capture of level change and hold time of high or low level.
- 26-bit PWM function is supported, which can dynamically adjust the PWM duty cycle setting.

## 8.2 Introduction to TMRx

Table 8-1 TMR0 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_TMR0_CTRL_MOD | 0x40002000 | Mode set register | 0x02 |
| R8_TMR0_INTER_EN | 0x40002002 | Interrupt enable register | 0x00 |
| R8_TMR0_INT_FLAG | 0x40002006 | Interrupt flag register | 0x00 |
| R8_TMR0_FIFO_COUNT | 0x40002007 | FIFO count register | 0x0X |
| R32_TMR0_COUNT | 0x40002008 | Current count value register | 0x0XXXXXXX |
| R32_TMR0_CNT_END | 0x4000200C | Final count value set register | 0x0XXXXXXX |
| R32_TMR0_FIFO | 0x40002010 | FIFO register | 0x0XXXXXXX |

Table 8-2 TMR1 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_TMR1_CTRL_MOD | 0x40002400 | Mode set register | 0x02 |
| R8_TMR1_CTRL_DMA | 0x40002401 | DMA control register | 0x00 |
| R8_TMR1_INTER_EN | 0x40002402 | Interrupt enable register | 0x00 |
| R8_TMR1_INT_FLAG | 0x40002406 | Interrupt flag register | 0x00 |
| R8_TMR1_FIFO_COUNT | 0x40002407 | FIFO count register | 0x0X |
| R32_TMR1_COUNT | 0x40002408 | Current count value register | 0x0XXXXXXX |
| R32_TMR1_CNT_END | 0x4000240C | Final count value register | 0x0XXXXXXX |
| R32_TMR1_FIFO | 0x40002410 | FIFO register | 0x0XXXXXXX |
| R16_TMR1_DMA_NOW | 0x40002414 | Current buffer address of DMA | 0x0000XXXX |
| R16_TMR1_DMA_BEG | 0x40002418 | Start buffer address of DMA | 0x0000XXXX |
| R16_TMR1_DMA_END | 0x4000241C | End buffer address of DMA | 0x0000XXXX |

Table 8-3 TMR2 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_TMR2_CTRL_MOD | 0x40002800 | Mode set register | 0x02 |
| R8_TMR2_CTRL_DMA | 0x40002801 | DMA control register | 0x00 |
| R8_TMR2_INTER_EN | 0x40002802 | Interrupt enable register | 0x00 |
| R8_TMR2_INT_FLAG | 0x40002806 | Interrupt flag register | 0x00 |
| R8_TMR2_FIFO_COUNT | 0x40002807 | FIFO count register | 0x0X |
| R32_TMR2_COUNT | 0x40002808 | Current count value register | 0x0XXXXXXX |
| R32_TMR2_CNT_END | 0x4000280C | Final count value register | 0x0XXXXXXX |
| R32_TMR2_FIFO | 0x40002810 | FIFO register | 0x0XXXXXXX |
| R16_TMR2_DMA_NOW | 0x40002814 | Current buffer address of DMA | 0x0000XXXX |
| R16_TMR2_DMA_BEG | 0x40002818 | Start buffer address of DMA | 0x0000XXXX |
| R16_TMR2_DMA_END | 0x4000281C | End buffer address of DMA | 0x0000XXXX |

Table 8-4 TMR3 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_TMR3_CTRL_MOD | 0x40002C00 | Mode set register | 0x02 |
| R8_TMR3_INTER_EN | 0x40002C02 | Interrupt enable register | 0x00 |
| R8_TMR3_INT_FLAG | 0x40002C06 | Interrupt flag register | 0x00 |
| R8_TMR3_FIFO_COUNT | 0x40002C07 | FIFO count register | 0x0X |
| R32_TMR3_COUNT | 0x40002C08 | Current count value register | 0x0XXXXXXX |
| R32_TMR3_CNT_END | 0x40002C0C | Final count value set register | 0x0XXXXXXX |
| R32_TMR3_FIFO | 0x40002C10 | FIFO register | 0x0XXXXXXX |

Mode set register (R8_TMRx_CTRL_MOD) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | RB_TMR_CAP_EDGE | RW | Capture trigger mode selection in capture mode:<br>00: Not triggered;<br>01: Capture the time between any edge changes;<br>10: Capture the time between falling edges;<br>11: Capture the time between rising edges. In the count mode, select the edge of count:<br>00: Not sample count;<br>01: Count when sampling to any edge;<br>10: Count when sampling to falling edge;<br>11: Count when sampling to rising edge. | 00b |
| [7:6] | RB_TMR_PWM_REPEAT | RW | Data repetition selection in PWM mode:<br>00: Repeat once; 01: Repeat 4 times;<br>10: Repeat 8 times; 11: Repeat 16 times. | 00b |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_TMR_CAP_COUNT | RW | Sub-mode of RB_TMR_MODE_IN=1 input | 0 |

| | | | | |
|---|---|---|---|---|
| | | | mode:<br>1: Count mode; 0: Capture mode. | |
| 4 | RB_TMR_OUT_POLAR | RW | In PWM mode, output polarity set:<br>1: Default at high level, active low;<br>0: Default at low level, active high; | 0 |
| 3 | RB_TMR_OUT_EN | RW | Timer output enable:<br>1: Output enabled; 0: Output disabled. | 0 |
| 2 | RB_TMR_COUNT_EN | RW | Timer count enable:<br>1: Enable counting; 0: Disable counting. | 0 |
| 1 | RB_TMR_ALL_CLEAR | RW | Clear the FIFO/counter/interrupt flag of timer:<br>1: Force to empty and clear;<br>0: Not clear. | 1 |
| 0 | RB_TMR_MODE_IN | RW | Timer mode set:<br>1: Input mode (capture mode or count mode);<br>0: Timing mode or PWM mode. | 0 |

Interrupt enable register (R8_TMRx_INTER_EN) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:5] | Reserved | RO | Reserved | 000b |
| 4 | RB_TMR_IE_FIFO_OV | RW | FIFO overflow (FIFO is full in capture mode or FIFO is empty in PWM mode) interrupt enable:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 3 | RB_TMR_IE_DMA_END | RW | DMA end interrupt enable (only TMR1/2 support):<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 2 | RB_TMR_IE_FIFO_HF | RW | FIFO used more than half (FIFO>=4 in capture mode or FIFO<4 in PWM mode) interrupt enable:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 1 | RB_TMR_IE_DATA_ACT | RW | Data activation (In capture mode, it means that every time new data is captured. In PWM mode, it means that value triggers the effective level to end) interrupt enable:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 0 | RB_TMR_IE_CYC_END | RW | Cycle end (it refers to timeout in capture mode, and it refers to the end of cycle in PWM mode and timing mode) interrupt enable:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |

Interrupt flag register (R8_TMRx_INT_FLAG) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:5] | Reserved | RO | Reserved | 000b |
| 4 | RB_TMR_IF_FIFO_OV | RW1 | FIFO overflow (FIFO is full in capture mode or FIFO is empty in PWM mode) flag. Write 1 to reset:<br>1: Has overflowed;<br>0: Not overflowed. | 0 |
| 3 | RB_TMR_IF_DMA_END | RW1 | DMA end. Write 1 to reset:<br>1: Has completed;<br>0: Not completed. | 0 |
| 2 | RB_TMR_IF_FIFO_HF | RW1 | FIFO used more than half (FIFO>=4 in capture mode or FIFO<4 in PWM mode) flag. Write 1 to reset:<br>1: FIFO has been used more than half; 0: FIFO has not been used more than half. | 0 |
| 1 | RB_TMR_IF_DATA_ACT | RW1 | Data activation (it means that every time new data is captured in capture mode, and it means that value triggers the effective level to end in PWM mode) flag. Write 1 to reset:<br>1: Data generated/used;<br>0: Not generated/not used. | 0 |
| 0 | RB_TMR_IF_CYC_END | RW1 | Cycle end (it refers to timeout in capture mode, and it refer to the end of cycle in PWM mode and timing mode) flag. Write 1 to reset:<br>1: Timeout/end of cycle;<br>0: No timeout/ not end. | 0 |

FIFO count register (R8_TMRx_FIFO_COUNT) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_TMRx_FIFO_COUNT | RO | Data count in FIFO, the maximum value is 8. | 0x0X |

Current count value register (R32_TMRx_COUNT) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | R32_TMRx_COUNT | RO | Current count value of counter. | 0XXXXXXXh |

Final count value set register (R32_TMRx_CNT_END) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | R32_TMRx_CNT_END | RW | In timer mode, the number of clocks in a timing cycle;<br>In PWM mode, the total number of | 0XXXXXXXh |

| | | | | |
|---|---|---|---|---|
| | | | clocks in a PWM cycle;<br>Capture the number of timeout clocks in capture mode.<br>Only the lower 26 bits are valid, and the maximum value is 67108863.<br>In counting mode, final count value -2 (overflow).<br>*Note: With write operation on this register, the value of R32_TMRx_COUNT will be automatically cleared to 0.* | |

FIFO register (R32_TMRx_FIFO) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:0] | R32_TMRx_FIFO | RO/WO | FIFO data register, only the lower 26 bits are valid. | 0XXXXXXXh |

DMA control register (R8_TMRx_CTRL_DMA) (x=1/2) (only TMR1/2 support)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:3] | Reserved | RO | Reserved | 00000b |
| 2 | RB_TMR_DMA_LOOP | RW | DMA address loop enable bit:<br>1: Enable address loop;<br>0: Disable address loop.<br>If the DMA address loop is enabled, when the DMA address is added to the set end address, it will automatically loop to the start address set. | 0 |
| 1 | Reserved | RO | Reserved | 0 |
| 0 | RB_TMR_DMA_ENABLE | RW | DMA function enable bit:<br>1: DMA enabled.<br>0: DMA disabled. | 0 |

DMA current buffer address (R16_TMRx_DMA_NOW) (x=1/2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_TMRx_DMA_NOW | RO | Current address of DMA data buffer.<br>It can be used to calculate the number of conversions, and the calculation method is: COUNT=(TMR_DMA_NOW-TMR_DMA_BEG)/4. | XXXXh |

DMA start buffer address (R16_TMRx_DMA_BEG) (x=1/2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_TMRx_DMA_BEG | RW | The start address of DMA data buffer, only the lower 15 bits are valid, and the address | XXXXh |

| | | | | |
|---|---|---|---|---|
| | | | must be 4 bytes aligned. | |

DMA end buffer address (R16_TMRx_DMA_END) (x=1/2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_TMRx_DMA_END | RW | The end address of DMA data buffer (not included), only the lower 15 bits are valid, and the address must be 4 bytes aligned. | XXXXh |

## 8.3 Functional description and configuration

### 8.3.1 Timing and counting functions

Each timer of the chip supports the longest time interval of 2^26 clock cycles and performs an incremental count mode. If the system clock cycle is 32MHz, the longest time interval is: 31.25nS*2^26≈2S. Each timer has an independent interrupt.

The operation steps for timing function are as follows:

(1) Set RB_TMR_ALL_CLEAR, clear R32_TMRx_COUNT and interrupt flag, etc.

(2) Set the R32_TMRx_CNT_END register to the time value that needs timing; Time = Tsys *R32_TMRx_CNT_END;

(3) Clear RB_TMR_ALL_CLEAR, clear the timing mode corresponding to RB_TMR_MODE_IN;

(4) Optional steps, set R8_TMRx_INTER_EN register, set RB_TMR_IE_CYC_END to open the timing cycle interrupt;

(5) Set the RB_TMR_COUNT_EN in the R8_TMRx_CTRL_MOD register, and start the timer counting;

(6) When count value of R32_TMRx_COUNT is equal to that of R32_TMRx_CNT_END, the timing is completed. In this case, RB_TMR_IF_CYC_END in R8_TMRx_INT_FLAG is set to 1, which can be cleared by writing 1.

The operation steps for counting function are as follows:

(1) Set the corresponding I/O pin direction of counting as input;

(2) Set the count overflow final value in R32_TMRx_CNT_END;

(3) Configure R8_TMRx_CTRL_MOD, set the corresponding count mode of RB_TMR_MODE_IN and RM_TMR_CAP_COUNT, clear RB_TMR_ALL_CLEAR, select sample edge method by RB_TMR_CAP_EDGE, set RB_TMR_COUNT_EN in R8_TMRx_CTRL_MOD to 1, enable counting function;

(4) Optional, set the corresponding interrupt enable register bit if it is needed to enable interrupt;

(5) Save current count value in R32_TMRx_COUNT. Every time the count value reaches final count value, RB_TMR_IE_CYC_END will be set to 1 and R32_TMRx_COUNT will be cleared to 0. Hardware interrupt is triggered if enabling interrupt.

### 8.3.2 PWM function

Each timer of the chip has PWM function. The PWM functions of TMR1 and TMR2 support DMA data loading. The default output polarity of PWM can be set to high level or low level. The repeated output times of the same data can be selected as 1, 4, 8 or 16. This repeat function is combined with DMA to simulate the effect of DAC. The shortest time unit for PWM to output valid level is 1 system clock cycle, and the duty cycle of PWM can be dynamically modified to simulate special waveforms.

PWM operation steps are as follows:

(1) Set RB_TMR_ALL_CLEAR, empty and clear R32_TMRx_FIFO and interrupt flags, etc.

(2) Set the PWM total cycle register R32_TMRx_CNT_END, the value shall not be less than the value in R32_TMRx_FIFO register;

(3) Configure R8_TMRx_CTRL_MOD, clear RB_TMR_ALL_CLEAR, clear PWM mode corresponding to RB_TMR_MODE_IN, select the output polarity through RB_TMR_OUT_POLAR, and select the repetition times of the same data through RB_TMR_PWM_REPEAT as needed;

(4) Set the data register R32_TMRx_FIFO, the minimum value is 0, with the corresponding duty cycle of 0%; the maximum value is the same as that of R32_TMR_CNT_END, with the corresponding duty cycle of 100%; the calculation of duty cycle: R32_TMRx_FIFO/R32_TMRx_CNT_END. TMR1 and TMR2 can load continuous dynamic data through DMA, and simulate special waveforms combined with the repeated output times of the same data;

(5) Configure R8_TMRx_CTRL_MOD, set RB_TMR_COUNT_EN to start counting and RB_TMR_OUT_EN to allow PWM output;

(6) Set the I/O pin corresponding to PWM as output;

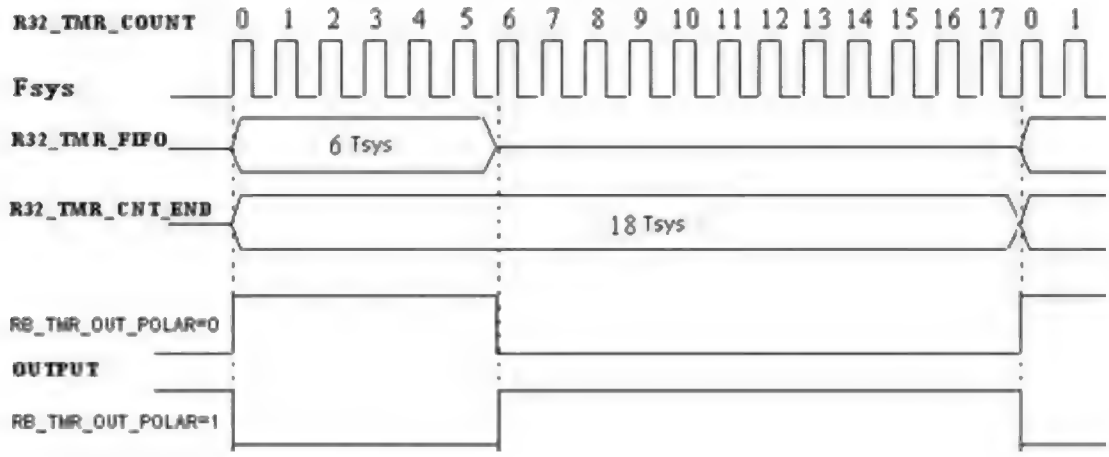
(7) Optional. If it is needed to enable interrupts, set the corresponding interrupt enable register bit;

(8) After a PWM cycle is completed, if an interrupt is enabled, the hardware interrupt will be triggered after RB_TMR_IF_DATA_ACT or RB_TMR_IF_CYC_END is set;

(9) The duty cycle of PWM can be dynamically changed by updating the data in R32_TMRx_FIFO. It is recommended to load it through DMA.

For example: Set the RB_TMR_OUT_POLAR bit to 0, R32_TMRx_FIFO to 6, R32_TMRx_CNT_END to 18, the basic timing diagram of PWM generation is as follows, and its duty cycle is: R32_TMRx_FIFO/R32_TMRx_CNT_END = 1/3.

Figure 8-1 PWM output timing diagram




If RB_TMR_PWM_REPEAT is set to 00, it means that the above process is repeated once, 01 means repeating 4 times, 10 means that repeating 8 times, and 11 means repeating 16 times. After repeating, load the next data in FIFO and then continue.

### 8.3.3 Capture function

Each timer of chip has a capture function, among which the capture functions of TMR1 and TMR2 support DMA data storage. 3 capture modes can be selected: start from any edge trigger and end at any edge trigger, start from

rising edge trigger and end at rising edge trigger, and start from falling edge trigger and end at falling edge trigger. The following table shows the description of capture trigger mode:

Table 8-5 Description of capture trigger mode

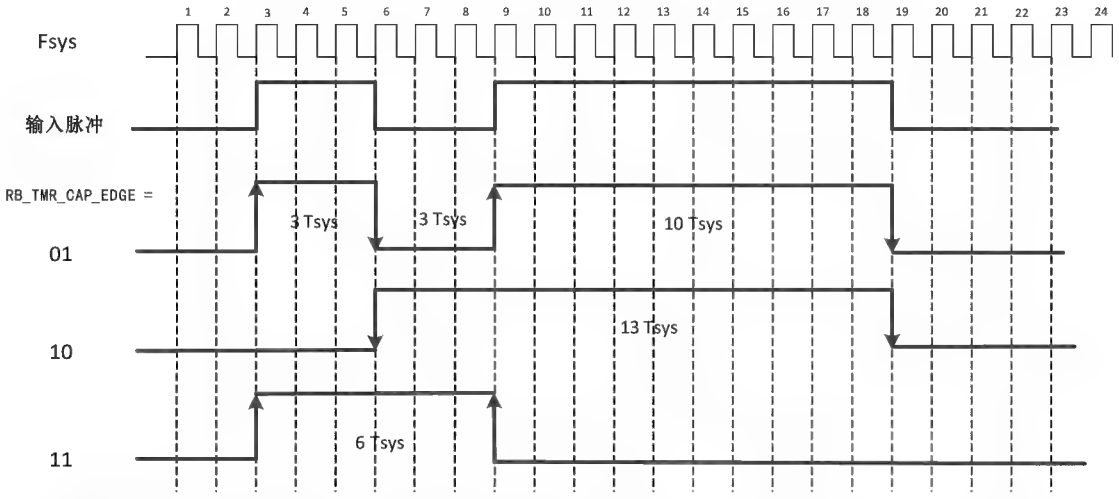
| Capture mode select bit RB_TMR_CATCH_EDGE | Trigger Mode | Icon |
|---|---|---|
| 00 | Not capture | None |
| 01 | Edge trigger edge to edge | |
| 10 | Falling edge to falling edge | |
| 11 | Rising edge to rising edge | |

There are 2 trigger states in edge trigger mode, which can capture high level width or low-level width. When the highest bit (bit 25) of the valid data in data register R32_TMRx_FIFO is 1, high level is captured; when it is 0, low level is captured. If the bit 25 of consecutive sets of data is 1 (or 0), the width of the high (or low) level exceeds the timeout value, and needs to be combined and accumulated.

In the trigger modes from falling edge to falling edge and from rising edge to rising edge, an input change cycle can be captured. When the highest bit (bit 25) of the valid data in data register R32_TMRx_FIFO is 0, one cycle is normally sampled; when it is 1, the input change period exceeds the timeout value R32_TMRx_CNT_END, and the latter set of data needs to be added and accumulated as a single input change period.

The specific description is shown in the figure below:

Figure 8-2 Count the capture cycle with system clock




As shown in the figure above, sample once in each clock cycle:

When RB_TMR_CATCH_EDGE=01b, sampling is set to the mode of edge trigger, and the time width sampled is 3, 3, 10;

When RB_TMR_CATCH_EDGE=10b, sampling is set to the mode of falling edge to falling edge, and the time width sampled is 13;

When RB_TMR_CATCH_EDGE=11b, sampling is set to the mode of rising edge to rising edge, and the time width sampled is 6.

Operation steps for capture mode:

(1) Set RB_TMR_ALL_CLEAR, empty and clear R32_TMRx_FIFO and interrupt flags, etc.
(2) Set the direction of the I/O pin corresponding to capture as input;
(3) Set a reasonable capture timeout time in R32_TMRx_CNT_END, which can be used to generate a timeout interrupt when the input signal remains unchanged for a long time, and generate timeout data after the input signal does not change overtime (bit 25 of data is 1, and the lower 25 bits can be accumulated backward);
(4) Configure R8_TMRx_CTRL_MOD, set the capture mode corresponding to RB_TMR_MODE_IN, select the edge mode of capture through RB_TMR_CAP_EDGE, set RB_TMR_COUNT_EN of R8_TMRx_CTRL_MOD as 1, and enable counting;
(5) Optional step: If it is needed to enable interrupts, set the corresponding interrupt enable register bit;
(6) To save the captured data in the way of DMA, you need to set the register R16_TMRx_DMA_BEG as the first address of buffer which stores the data captured, set the register R16_TMRx_DMA_END as the end address of buffer which stores the data captured (not included), and set the RB_TMR_DMA_ENABLE of R8_TMRx_CTRL_DMA as 1, and enable DMA function;
(7) Clear RB_TMR_ALL_CLEAR of R8_TMRx_CTRL_MOD, and start the capture function;
(8) Every time data is captured, RB_TMR_IF_DATA_ACT will be set as 1; if the interrupt is enabled, a hardware interrupt will be triggered; the captured data is stored in R32_TMRx_FIFO by default; if DMA is enabled, the captured data will be automatically stored in the data buffer set by DMA.

# Chapter 9 Universal Asynchronous Receiver-Transmitter (UART)

## 9.1 Introduction to UART

CH583 and CH582 each provides 4 sets of full-duplex UARTs (UART0/1/2/3). Full-duplex and half-duplex serial communication are supported. Among them, UART0 provides the transmit status pin for switching RS485, and supports MODEM signals CTS, DSR, RI, DCD, DTR and RTS.

### 9.1.1 Alternate functions

- Compatible with 16C550 asynchronous serial port and enhanced.
- 5/6/7/8 data bits, 1/2 stop bits.
- Support the verification modes of odd, even, no parity, blank 0 and flag 1, etc.
- Programmable communication baud rate, up to 6Mbps.
- Built-in 8-byte FIFO buffer, support 4 FIFO trigger stages.
- UART0 supports MODEM signals CTS, DSR, RI, DCD, DTR and RTS.
- UART0 supports automatic handshake and automatic transmission rate control of hardware flow control signals CTS and RTS, compatible with TL16C550C.
- Support serial frame error detection and Break circuit interval detection.
- Full-duplex and half-duplex serial communication are supported, and UART0 provides the transmit status pin for switching RS485.

## 9.2 Register description

Table 9-1 UART0 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UART0_MCR | 0x40003000 | MODEM control register | 0x00 |
| R8_UART0_IER | 0x40003001 | Interrupt enable register | 0x00 |
| R8_UART0_FCR | 0x40003002 | FIFO control register | 0x00 |
| R8_UART0_LCR | 0x40003003 | Line control register | 0x00 |
| R8_UART0_IIR | 0x40003004 | Interrupt identification register | 0x01 |
| R8_UART0_LSR | 0x40003005 | Line status register | 0x60 |
| R8_UART0_MSR | 0x40003006 | MODEM status register | 0xX0 |
| R8_UART0_RBR | 0x40003008 | Receive buffer register | 0xXX |
| R8_UART0_THR | 0x40003008 | Transmit hold register | 0xXX |
| R8_UART0_RFC | 0x4000300A | Receive FIFO count register | 0x00 |
| R8_UART0_TFC | 0x4000300B | Transmit FIFO count register | 0x00 |
| R16_UART0_DL | 0x4000300C | Baud rate divisor latch | 0xXX |
| R8_UART0_DIV | 0x4000300E | Prescaler divisor register | 0xXX |
| R8_UART0_ADR | 0x4000300F | Slave address register | 0xFF |

Table 9-2 UART1 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UART1_MCR | 0x40003400 | MODEM control register | 0x00 |
| R8_UART1_IER | 0x40003401 | Interrupt enable register | 0x00 |
| R8_UART1_FCR | 0x40003402 | FIFO control register | 0x00 |
| R8_UART1_LCR | 0x40003403 | Line control register | 0x00 |
| R8_UART1_IIR | 0x40003404 | Interrupt identification register | 0x01 |
| R8_UART1_LSR | 0x40003405 | Line status register | 0x60 |
| R8_UART1_RBR | 0x40003408 | Receive buffer register | 0xXX |
| R8_UART1_THR | 0x40003408 | Transmit hold register | 0xXX |
| R8_UART1_RFC | 0x4000340A | Receive FIFO count register | 0x00 |
| R8_UART1_TFC | 0x4000340B | Transmit FIFO count register | 0x00 |
| R16_UART1_DL | 0x4000340C | Baud rate divisor latch | 0xXX |
| R8_UART1_DIV | 0x4000340E | Prescaler divisor register | 0xXX |

Table 9-3 UART2 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UART2_MCR | 0x40003800 | MODEM control register | 0x00 |
| R8_UART2_IER | 0x40003801 | Interrupt enable register | 0x00 |
| R8_UART2_FCR | 0x40003802 | FIFO control register | 0x00 |
| R8_UART2_LCR | 0x40003803 | Line control register | 0x00 |
| R8_UART2_IIR | 0x40003804 | Interrupt identification register | 0x01 |
| R8_UART2_LSR | 0x40003805 | Line status register | 0x60 |
| R8_UART2_RBR | 0x40003808 | Receive buffer register | 0xXX |
| R8_UART2_THR | 0x40003808 | Transmit hold register | 0xXX |
| R8_UART2_RFC | 0x4000380A | Receive FIFO count register | 0x00 |
| R8_UART2_TFC | 0x4000380B | Transmit FIFO count register | 0x00 |
| R16_UART2_DL | 0x4000380C | Baud rate divisor latch | 0xXX |
| R8_UART2_DIV | 0x4000380E | Prescaler divisor register | 0xXX |

Table 9-4 UART3 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UART3_MCR | 0x40003C00 | MODEM control register | 0x00 |
| R8_UART3_IER | 0x40003C01 | Interrupt enabling register | 0x00 |
| R8_UART3_FCR | 0x40003C02 | FIFO control register | 0x00 |
| R8_UART3_LCR | 0x40003C03 | Line control register | 0x00 |
| R8_UART3_IIR | 0x40003C04 | Interrupt identification register | 0x01 |
| R8_UART3_LSR | 0x40003C05 | Line status register | 0x60 |
| R8_UART3_RBR | 0x40003C08 | Receive buffer register | 0xXX |

| | | | |
|---|---|---|---|
| R8_UART3_THR | 0x40003C08 | Transmit hold register | 0xXX |
| R8_UART3_RFC | 0x40003C0A | Receive FIFO count register | 0x00 |
| R8_UART3_TFC | 0x40003C0B | Transmit FIFO count register | 0x00 |
| R16_UART3_DL | 0x40003C0C | Baud rate divisor latch | 0xXX |
| R8_UART3_DIV | 0x40003C0E | Prescaler divisor register | 0xXX |

MODEM control register (R8_UARTx_MCR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_MCR_HALF | RW | Half-duplex transceiver mode control (only supported by UART0):<br>1: Enter half-duplex transceiver mode, transmit with priority, and receive when not transmitting;<br>0: Disable half-duplex mode. | 0 |
| 6 | RB_MCR_TNOW | RW | Status enable that DTR pin output is being transmitted (only supported by UART0):<br>1: Output the indication status of being transmitted to DTR pin, used to control the RS485 receive/transmit switch.<br>0: DTR pin is in normal function. | 0 |
| 5 | RB_MCR_AU_FLOW_EN | RW | CTS and RTS hardware automatic flow control enable (only supported by UART0):<br>1: Enable;<br>0: Disable;<br>In the flow control mode, if this bit is 1, then UART will continue to send the next data only when it detects that the CTS pin input is valid (active low). Otherwise, the UART transmission will be suspended, and the CTS input status change will not generate MODEM status interrupt when this bit is 1. If this bit is 1 and RTS is 1, UART will automatically validate the RTS pin (active low) when receiver FIFO is empty. UART will automatically invalidate the RTS pin when the number of received bytes reaches the trigger point of FIFO and will re-validate the RTS pin when the receiver FIFO is empty. Hardware automatic flow control can be used to connect your own CTS pin to the other party's RTS pin and transmit your own RTS pin to the other party's CTS pin. | 0 |

| | | | | |
|---|---|---|---|---|
| 4 | RB_MCR_LOOP | RW | Test mode of internal loop enable (only supported by UART0):<br>1: Enable the test mode of internal loop;<br>0: Disable the test mode of internal loop.<br>In the test mode of the internal loop, all external output pins of the serial port are invalid, TXD internally returns to RXD, RTS internally returns to CTS, DTR internally returns to DSR, OUT1 internally returns to RI and OUT2 internally returns to DCD. | 0 |
| 3 | RB_MCR_OUT2<br>RB_MCR_INT_OE | RW | UART interrupt request output control:<br>1: Enable to send request; 0: Disable. | 0 |
| 2 | RB_MCR_OUT1 | RW | User-defined MODEM control (only supported by UART0), and no actual output pin is connected:<br>1: Set high; 0: Set low. | 0 |
| 1 | RB_MCR_RTS | RW | RTS signal output level control (only supported by UART0):<br>1: RTS signal output is valid (low level);<br>0: RTS signal output high level (default). | 0 |
| 0 | RB_MCR_DTR | RW | DTR signal output level control (only supported by UART0):<br>1: DTR signal output is valid (low level);<br>0: DTR signal output high level (default). | 0 |

Interrupt enable register (R8_UARTx_IER) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_IER_RESET | WZ | UART software reset control bit, automatically cleared:<br>1: Software resets UART;<br>0: Normal operation. | 0 |
| 6 | RB_IER_TXD_EN | RW | UART TXD pin output enable bit:<br>1: Enable pin output;<br>0: Disable pin output. | 0 |
| 5 | RB_IER_RTS_EN | RW | RTS pin output enable bit (only supported by UART0):<br>1: Enable output; 0: Disable output. | 0 |
| 4 | RB_IER_DTR_EN | RW | DTR pin output enable bit (only supported by UART0):<br>1: Enable output; 0: Disable output. | 0 |
| 3 | RB_IER_MODEM_CHG | RW | Modem input status change interrupt enable bit (only supported by UART0):<br>1: Enable interrupt; 0: Disable interrupt. | 0 |

| | | | | |
|---|---|---|---|---|
| 2 | RB_IER_LINE_STAT | RW | Receive line status interrupt enable bit:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 1 | RB_IER_THR_EMPTY | RW | Transmit hold register empty interrupt enable bit:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 0 | RB_IER_RECV_RDY | RW | Receive data interrupt enable bit:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |

FIFO control register (R8_UARTx_FCR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | RB_FCR_FIFO_TRIG | RW | Trigger points select of receiving FIFO interrupt and hardware flow control:<br>00: 1 byte; 01: 2 bytes;<br>10: 4 bytes; 11: 7 bytes.<br>Used to set the trigger points of receiving FIFO interrupt and hardware flow control. For example: 10 corresponds to 4 bytes, that is, interrupt available for receiving data is generated when 4 bytes are received, and RTS pin is automatically invalidated when hardware flow control is enabled. | 00b |
| [5:3] | Reserved | RO | Reserved | 000b |
| 2 | RB_FCR_TX_FIFO_CLR | WZ | Transmit FIFO data clear enable bit, and automatically cleared:<br>1: Clear the data of transmitter FIFO (excluding TSR);<br>0: Not clear the data of transmitter FIFO. | 0 |
| 1 | RB_FCR_RX_FIFO_CLR | WZ | Receive FIFO data clear enable bit, and automatically cleared:<br>1: Clear the data of receiver FIFO (excluding RSR);<br>0: Not clear the data of receiver FIFO. | 0 |
| 0 | RB_FCR_FIFO_EN | RW | FIFO enable bit:<br>1: Enable 8-byte FIFO; 0: Disable FIFO.<br>After disabling FIFO, it is 16C450 compatible mode, which means that there is only 1 byte in FIFO (RECV_TG1=0, RECV_TG0=0, FIFO_EN=1), and it is recommended to enable. | 0 |

Line control register (R8_UARTx_LCR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_LCR_DLAB<br>RB_LCR_GP_BIT | RW | UART general purpose bit, user-defined. | 0 |

| | | | | |
|---|---|---|---|---|
| 6 | RB_LCR_BREAK_EN | RW | Force to generate BREAK line interval enable:<br>1: Forced to generate;<br>0: Not generate. | 0 |
| [5:4] | RB_LCR_PAR_MOD | RW | Parity bit format selection:<br>00: Odd;<br>01: Even;<br>10: Mark (MARK, set to 1);<br>11: Space (SPACE, cleared).<br>Valid only when RB_LCR_PAR_EN is 1. | 00b |
| 3 | RB_LCR_PAR_EN | RW | Parity bit enable:<br>1: Allow to generate parity bit when sending and check parity bit when receiving;<br>0: No parity bit. | 0 |
| 2 | RB_LCR_STOP_BIT | RW | Stop bit format set:<br>0: 1 stop bit; 1: 2 stop bits. | 0 |
| [1:0] | RB_LCR_WORD_SZ | RW | UART data length selection:<br>00: 5 data bits; 01: 6 data bits;<br>10: 7 data bits; 11: 8 data bits. | 00b |

Interrupt identification register (R8_UARTx_IIR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | RB_IIR_FIFO_ID | RO | UART FIFO enable status:<br>11: FIFO has been enabled;<br>00: FIFO is not enabled. | 00b |
| [5:4] | Reserved | RO | Reserved | 00b |
| [3:0] | RB_IIR_INT_MASK | RO | Interrupt flag: If the RB_IIR_NO_INT bit is 0, an interrupt is generated, and it is needed to judge the interrupt source after reading. Please refer to Table 9-5 for details. | 0000b |
| 0 | RB_IIR_NO_INT | RO | UART no interrupt flag:<br>1: No interrupt; 0: Interrupt. | 1 |

The meanings of bit RB_IIR_NO_INT of interrupt identification register R8_UARTx_IIR and each bit of RB_IIR_INT_MASK is shown in the following table:

Table 9-5 Meaning of RB_IIR_INT_MASK in IIR register

| IIR register bit | | | | Priority | Interrupt type | Interrupt source | Means of clearing interrupts |
|---|---|---|---|---|---|---|---|
| IID3 | IID2 | IID1 | NOINT | | | | |
| 0 | 0 | 0 | 1 | None | No interrupt | No interrupt | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 0 | Bus address matching | The received one data is the UART bus address, and the address matches the preset slave value or the broadcast address. (Only supported by UART0) | Read IIR or disable multi-device mode |
| 0 | 1 | 1 | 0 | 1 | Receive line status | OVER_ERR/PAR_ERR/FRAM_ERR/BREAK_ERR | Read LSR |
| 0 | 1 | 0 | 0 | 2 | Receive data available | The number of bytes received reaches the trigger point of FIFO. | Read RBR |
| 1 | 1 | 0 | 0 | 2 | Receive data timeout | The next data is not received for more than 4 data periods. | Read RBR |
| 0 | 0 | 1 | 0 | 3 | THR register empty | Transmit hold register is empty, or RB_IER_THR_EMPTY bit is changed from 0 to 1 and triggered. | Read IIR or write THR |
| 0 | 0 | 0 | 0 | 4 | MODEM input change | Triggered by setting △ CTS, △ DSR, △ RI and △ DCD to 1. | Read MSR |

Line status register (R8_UARTx_LSR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_LSR_ERR_RX_FIFO | RO | Receive FIFO error flag:<br>1: At least one PAR_ERR or FRAM_ERR or BREAK_ERR error in the receiver FIFO;<br>0: No error in receiver FIFO. | 0 |
| 6 | RB_LSR_TX_ALL_EMP | RO | THR and TSR empty flag.<br>1: Both are empty;<br>0: Both are not empty. | 1 |
| 5 | RB_LSR_TX_FIFO_EMP | RO | Transmit FIFO empty flag:<br>1: Transmit FIFO is empty;<br>0: Transmit FIFO is not empty. | 1 |
| 4 | RB_LSR_BREAK_ERR | RZ | BREAK line interval detection flag:<br>1: BREAK is detected;<br>0: BREAK is not detected. | 0 |
| 3 | RB_LSR_FRAME_ERR | RZ | Data frame error flag:<br>1: Frame error in the data being read from the receiver FIFO, and a valid stop bit is missing.<br>0: No error in the currently read data frame. | 0 |
| 2 | RB_LSR_PAR_ERR | RZ | Receive data Parity error flag:<br>1: Parity error in the data being read from the receiver FIFO.<br>0: The currently read data parity is correct. | 0 |
| 1 | RB_LSR_OVER_ERR | RZ | Receiver FIFO buffer overflow flag:<br>1: Has overflowed; 0: Not overflowed. | 0 |
| 0 | RB_LSR_DATA_RDY | RO | Receiver FIFO receive data flag:<br>1: Data in FIFO; 0: No data.<br>After reading all the data in the FIFO, this bit will | 0 |

| | | | | |
|---|---|---|---|---|
| | | | be automatically cleared. | |

MODEM status register (R8_UART0_MSR) (only supported by UART0)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_MSR_DCD | RO | DCD pin status:<br>1: DCD is active (low level);<br>0: DCD is inactive (high level). | 0 |
| 6 | RB_MSR_RI | RO | RI pin status:<br>1: RI is active (low level);<br>0: RI is inactive (high level). | 0 |
| 5 | RB_MSR_DSR | RO | DSR pin status:<br>1: DSR pin is active (low level);<br>0: DSR pin is inactive (high level). | X |
| 4 | RB_MSR_CTS | RO | CTS pin status:<br>1: CTS pin is active (low level);<br>0: CTS pin is inactive (high level). | X |
| 3 | RB_MSR_DCD_CHG | RZ | DCD input status change flag:<br>1: Has changed; 0: No change. | 0 |
| 2 | RB_MSR_RI_CHG | RZ | RI input status change flag:<br>1: Has changed; 0: No change. | 0 |
| 1 | RB_MSR_DSR_CHG | RZ | DSR pin input status change flag:<br>1: Has changed; 0: No change. | 0 |
| 0 | RB_MSR_CTS_CHG | RZ | CTS pin input status change flag:<br>1: Has changed; 0: No change. | 0 |

Receive buffer register (R8_UARTx_RBR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UARTx_RBR | RO | Data receive buffer register.<br>If the DATA_RDY bit of LSR is 1, the received data can be read from this register;<br>If FIFO_EN is 1, the data received from UART shift register RSR will be firstly stored in the receiver FIFO, and then read out through the register. | XXh |

Transmit hold register (R8_UARTx_THR) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UARTx_THR | WO | Transmit hold register.<br>Transmitter FIFO is included, used to write the data to be transmitted; if FIFO_EN is 1, the written data will be firstly stored in the transmitter FIFO, and then output one by one through the transmit shift register TSR. | XXh |

Receive FIFO count register (R8_UARTx_RFC) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UARTx_RFC | RO | Data count in the current receiver FIFO. | 00h |

Transmit FIFO count register (R8_UARTx_TFC) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UARTx_TFC | RO | Data count in the current transmitter FIFO. | 00h |

Baud rate divisor latch (R16_UARTx_DL) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_UARTx_DL | RW | The 16-bit divisor is used to calculate the baud rate.<br>Formula: Divisor = the serial internal reference clock Fuart / 16 / the required communication baud rate.<br>For example: If the serial internal reference clock Fuart is 1.8432MHz and the required baud rate is 9600bps, then the divisor =1843200/16/9600=12. | XXXXh |

Prescaler divisor register (R8_UARTx_DIV) (x=0/1/2/3)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UARTx_DIV | RW | It is used to calculate the internal reference clock of UART, the lower 7 bits are valid.<br>Formula: Divisor = Fsys*2 / internal reference clock of UART, the maximum value is 127. | XXh |

Slave address register (R8_UART0_ADR) (only supported by UART0)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UART0_ADR | RW | Slave address of UART0 during multi-device communication:<br>FFh: Not used;<br>Others: Slave address. | 0FFh |

R8_UART0_ADR presets the address when this device acts as a slave, it is used to automatically compare the received addresses during multi-device communication, and generate an interrupt when the address matches or when the broadcast address 0FFH is received. Meanwhile, it is allowed to receive subsequent data packets. Data is not received until the address matches. After it starts sending data or rewriting the R8_UART0_ADR register, stop receiving any data, until the address matches again next time or the broadcast address is received.

When R8_UART0_ADR is 0FFH or RB_LCR_PAR_EN=0, the automatic comparison function of bus address is disabled.

When R8_UART0_ADR is not 0FFH and RB_LCR_PAR_EN=1, the automatic comparison function of bus address is enabled, and the following parameters should be configured: RB_LCR_WORD_SZ is 11b to select method of 8 data bits. For the case when the address byte is MARK (that is, the bit 9 of data byte is 0), RB_LCR_PAR_MOD should be set to 10b. For the case when the address byte is SPACE (that is, the bit 9 of data byte is 1), RB_LCR_PAR_MOD should be set to 11b.

## 9.3 Functional description and configuration

UART0/1/2/3 output pins are all at 3.3V LVCMOS level. The pins in asynchronous serial port mode include: data transmission pins and MODEM contact signal pins (only supported by UART0). Data transmission pins include: TXD pin and RXD pin, both of which are at high level by default. MODEM contact signal pins include: CTS pin, DSR pin, RI pin, DCD pin, DTR pin, RTS pin, all of which are at high level by default.

All these MODEM contact signals can be used as general-purpose I/O pins, and the application program controls them and define their purposes.

4 sets of UARTs have built-in independent transceiver buffers and 8-byte FIFOs, support simplex, half-duplex or full-duplex UART communication. Serial data includes 1 low-level start bit, 5/6/7/8 data bits, 0/1 additional check bit or flag bit, 1/2 high-level stop bits, and supports odd/even/mark/blank checking. The baud rate error of the serial port transmitting signal is less than 0.5%, and the allowable baud rate error of the serial port receiving signal is not more than 2%.

### 9.3.1 Baud rate calculation

1) Calculate the internal reference clock Fuart of serial port, set the R8_UARTx_DIV register, the maximum value is 127, and usually 1 is written.
2) Calculate the baud rate and set R16_UARTx_DL register.
   Baud rate =Fsys * 2 / R8_UARTx_DIV / 16 / R16_UARTx_DL 。

### 9.3.2 UART transmission

"THR register empty" interrupt UART_II_THR_EMPTY sent by UART means that the current transmitter FIFO is empty. The interrupt is cleared when the IIR register is read, or cleared when the next data is written to THR. If only one byte is written to THR, it will soon generate again a request to send THR register empty interrupt as the byte is quickly transferred to the transmitter shift register (TSR) to start transmitting. At this point, the next data ready to be transmitted can be written. After all the data in TSR register is removed, UART transmission is completed. At this time, RB_LSR_TX_ALL_EMP bit of LSR register becomes active at 1.

In interrupt trigger mode, when THR empty interrupt from UART is received, if FIFO is enabled, up to 8 bytes can be written to THR and FIFO at a time, and then will be transmitted automatically by the controller in sequence; if FIFO is disabled, only one byte can be written at a time. If no data needs to be transmitted, exit directly (the interrupts have been automatically cleared when IIR is read before).

In the query mode, whether the transmitter FIFO is empty can be judged according to RB_LSR_TX_FIFO_EMP bit of LSR. When this bit is 1, the data can be written to THR and FIFO. If FIFO is enabled, up to 8 bytes can be written at a time.

R8_UARTx_TFC register can also be read to determine the number of remaining data to be sent in the current FIFO. If it is not equal to 8, continue to write the data to be sent into the FIFO, and that can save filling time.

### 9.3.3 UART transmission

UART receive data available interrupt UART_II_RECV_RDY means that the number of existing data bytes in the receiver FIFO has reached or exceeded the FIFO trigger points set and selected by RB_FCR_FIFO_TRIG of FCR. The interrupt is cleared when the data is read from RBR to cause the number of bytes in the FIFO less than that of the FIFO trigger points.

UART receive data timeout interrupt UART_II_RECV_TOUT means that there is at least 1-byte data in the receiver FIFO, and the waiting time is equivalent to the time of receiving 4 data starting from the last time when UART receives data and the last time when the system takes the data. The interrupt is cleared when a new data is received again or after the MCU reads RBR once. When receiver FIFO is empty, RB_LSR_DATA_RDY bit of LSR is 0; when there is data in the receiver FIFO, it is valid when RB_LSR_DATA_RDY bit is 1.

In the interrupt trigger mode, R8_UARTx_RFC register can be read to query the remaining data count in the current FIFO after receiving UART receive data timeout interrupt, and read all the data directly, or continuously query the RB_LSR_DATA_RDY of LSR. If this bit is valid, read the data until this bit becomes invalid. After receiving UART receive data available interrupt, read the data for the number of bytes set by RB_FCR_FIFO_TRIG from RBR at one time, or read all the data in the current FIFO according to the RB_LSR_DATA_RDY bit and the R8_UARTx_RFC register.

In query mode, whether the receiver FIFO is empty can be judged according to the RB_LSR_DATA_RDY bit of LSR, or read the R8_UARTx_RFC register to get the data count in the current FIFO and get all the data received by UART.

### 9.3.4 Hardware flow control

Hardware flow control includes automatic CTS (RB_MCR_AU_FLOW_EN is set to 1) and automatic RTS (RB_MCR_AU_FLOW_EN and RB_MCR_RTS are both set to 1).

If automatic CTS is enabled, CTS pin must be active before UART sends data. The serial port transmitter detects CTS pin before sending the next data. When the CTS pin is active, the transmitter sends the next data. In order to ensure that the transmitter stops sending the later data, CTS pin must be disabled before the middle time of the last stop bit currently being sent. The automatic CTS function reduces the interrupt applied to the system. When hardware flow control is enabled, a change in CTS pin level does not trigger a MODEM interrupt as the controller automatically controls the transmitter based on CTS pin status. If automatic RTS is enabled, RTS pin output will be valid only when there is enough space in FIFO to receive data, and RTS pin output is invalid when the receiver FIFO is full. RTS pin output will be valid if all the data in the receiver FIFO is taken or cleared. When the trigger point of the receiver FIFO is reached (the number of existing bytes in the receiver FIFO is not less than the number of bytes set by RB_FCR_FIFO_TRIG of FCR), RTS pin output is invalid, and the other transmitter is allowed to send another data after RTS pin is inactive. Once the data in the receiver FIFO is emptied, RTS pin will be automatically re-enabled, so that the other transmitter resumes transmission. If both automatic CTS and automatic RTS are enabled (both RB_MCR_AU_FLOW_EN and RB_MCR_RTS of MCR register are 1), when its own RTS pin is connected to the other CTS pin, the other side will not send data unless there is enough space in the receiver FIFO of the other side. Therefore, with hardware flow control, FIFO overflow and timeout errors during serial port reception can be avoided.

# Chapter 10 Serial Peripheral Interface (SPI)

## 10.1 Introduction to UART

SPI is a full-duplex serial interface with a host and several slaves connected to the bus, and only a pair of host and slave is communicating at the same time. Usually, SPI interface consists of 4 pins: SPI chip selected pin SCS, SPI clock pin (SCK), SPI serial data pin MISO (master input/slave output pin) and SPI serial data pin MOSI (master output/slave input pin).

### 10.1.1 Main features

CH583 provides 2 SPI interfaces. CH582 each provides only SPI0.

- SPI0 supports both master mode and slave mode. SPI1 only supports master mode.
- Compatible with Serial Peripheral Interface (SPI) specification.
- Data transfer modes: mode0 and mode3.
- 8-bit data transmission mode, optional data bit sequence: low bits of a byte are in front or high bits are in front.
- Clock frequency can be up to half of the system clock frequency (Fsys).
- 8-byte FIFO.
- SPI0 slave mode supports the first byte as command mode or data stream mode.
- SPI0 supports DMA, so the data transmission efficiency is higher.

## 10.2 Register description

Table 10-1 SPI0 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_SPI0_CTRL_MOD | 0x40004000 | SPI0 mode control register | 0x02 |
| R8_SPI0_CTRL_CFG | 0x40004001 | SPI0 configuration register | 0x00 |
| R8_SPI0_INTER_EN | 0x40004002 | SPI0 interrupt enable register | 0x00 |
| R8_SPI0_CLOCK_DIV<br>R8_SPI0_SLAVE_PRE | 0x40004003 | SPI0 clock divider register in master mode<br>SPI0 preset data register in slave mode | 0x10 |
| R8_SPI0_BUFFER | 0x40004004 | SPI0 data buffer | 0xXX |
| R8_SPI0_RUN_FLAG | 0x40004005 | SPI0 working status register | 0x00 |
| R8_SPI0_INT_FLAG | 0x40004006 | SPI0 interrupt flag register | 0x40 |
| R8_SPI0_FIFO_COUNT | 0x40004007 | SPI0 transceiver FIFO count register | 0x00 |
| R16_SPI0_TOTAL_CNT | 0x4000400C | SPI0 data receive/transmit data total length register | 0x0000 |
| R8_SPI0_FIFO | 0x40004010 | SPI0 data FIFO register | 0xXX |
| R8_SPI0_FIFO_COUNT1 | 0x40004013 | SPI0 transceiver FIFO count register | 0x00 |
| R16_SPI0_DMA_NOW | 0x40004014 | Current address of SPI0 DMA buffer | 0xXXXX |
| R16_SPI0_DMA_BEG | 0x40004018 | Start address of SPI0 DMA buffer | 0xXXXX |

| | | | |
|---|---|---|---|
| R16_SPI0_DMA_END | 0x4000401C | End address of SPI0 DMA buffer | 0xXXXX |

Table 10-2 SPI1 registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_SPI1_CTRL_MOD | 0x40004400 | SPI1 mode control register | 0x02 |
| R8_SPI1_CTRL_CFG | 0x40004401 | SPI1 configuration register | 0x00 |
| R8_SPI1_INTER_EN | 0x40004402 | SPI1 interrupt enable register | 0x00 |
| R8_SPI1_CLOCK_DIV | 0x40004403 | SPI1 clock frequency division register in master mode | 0x10 |
| R8_SPI1_BUFFER | 0x40004404 | SPI1 data buffer | 0xXX |
| R8_SPI1_RUN_FLAG | 0x40004405 | SPI1 working status register | 0x00 |
| R8_SPI1_INT_FLAG | 0x40004406 | SPI1 interrupt flag register | 0x40 |
| R8_SPI1_FIFO_COUNT | 0x40004407 | SPI1 transceiver FIFO count register | 0x00 |
| R16_SPI1_TOTAL_CNT | 0x4000440C | SPI1 receive/transmit data total length register | 0x00 |
| R8_SPI1_FIFO | 0x40004410 | SPI1 data FIFO register | 0xXX |
| R8_SPI1_FIFO_COUNT1 | 0x40004413 | SPI1 transceiver FIFO count register | 0x00 |

SPI mode control register (R8_SPIx_CTRL_MOD) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_SPI_MISO_OE | RW | MISO pin output enable (can be used at data line switching direction in 2-wire mode):<br>1: MISO output enabled;<br>0: MISO output disabled. | 0 |
| 6 | RB_SPI_MOSI_OE | RW | MOSI pin output enable:<br>1: MOSI output enabled;<br>0: MOSI output disabled. | 0 |
| 5 | RB_SPI_SCK_OE | RW | SCK pin output enable:<br>1: SCK output enabled;<br>0: SCK output disabled. | 0 |
| 4 | RB_SPI_FIFO_DIR | RW | FIFO direction:<br>1: Input (receive data);<br>0: Output (transmit data). | 0 |
| 3 | RB_SPI_SLV_CMD_MOD | RW | First byte mode selection in SPI0 slave mode (only supported by SPI0):<br>1: First byte command mode;<br>0: Data stream mode.<br>In the first byte command mode, it will be regarded as a command code when receiving the first byte of data after the SPI chip select is valid and RB_SPI_IF_FST_BYTE will be set to 1. | 0 |

| | | | | |
|---|---|---|---|---|
| 3 | RB_SPI_MST_SCK_MOD | RW | Clock idle mode selection in master mode:<br>1: Mode3 (SCK is at high level when idle);<br>0: Mode0 (SCK is at low level when idle). | 0 |
| 2 | RB_SPI_2WIRE_MOD | RW | 2-wire or 3-wire SPI mode selection in slave mode:<br>1: 2-wire mode/halfduplex (SCK/MISO);<br>0: 3-wire mode/full duplex (SCK/MOSI/MISO). | 0 |
| 1 | RB_SPI_ALL_CLEAR | RW | SPI FIFO/counter/interrupt flag clear:<br>1: Force to empty and clear;<br>0: Not clear. | 1 |
| 0 | RB_SPI_MODE_SLAVE | RW | SPI0 master/slave mode selection (only supported by SPI0):<br>1: Slave mode;<br>0: Master mode. | 0 |

SPI configuration register (R8_SPIx_CTRL_CFG) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| 6 | RB_SPI_MST_DLY_EN | RW | Input delay enable in master mode:<br>1: Enable, used for high-speed applications such as SPI clock close to half of Fsys;<br>0: Disable, regular applications. | 0 |
| 5 | RB_SPI_BIT_ORDER | RW | SPI data bit order selection:<br>1: LSB first,<br>0: MSB first. | 0 |
| 4 | RB_SPI_AUTO_IF | RW | Enable the function of automatically clearing flag bit RB_SPI_IF_BYTE_END when accessing BUFFER/FIFO:<br>1: Enable; 0: Disable. | 0 |
| 3 | Reserved | RO | Reserved | 0 |
| 2 | RB_SPI_DMA_LOOP | RW | DMA address loop enable:<br>1: Enable address loop; 0: Disable address loop.<br>If the DMA address loop is enabled, when the DMA address is added to the set end address, the auto loop points to the set first address. | 0 |
| 1 | Reserved | RO | Reserved | 0 |
| 0 | RB_SPI_DMA_ENABLE | RW | DMA enable (only supported by SPI0):<br>1: Enable; 0: Disable. | 0 |

SPI interrupt enable register (R8_SPIx_INTER_EN) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_SPI_IE_FST_BYTE | RW | In the first byte command mode of slave mode, first byte interrupt receive enable (only supported by SPI0): | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Enable receiving the first byte interrupt;<br>0: Disable receiving the first byte interrupt. | |
| [6：5] | Reserved | R0 | Reserved | 00b |
| 4 | RB_SPI_IE_FIFO_OV | RW | FIFO overflow (FIFO is full when receiving, or FIFO is empty when sending) interrupt enable (only supported by SPI0):<br>1: Interrupt enabled;<br>0: Interrupt disabled. | 0 |
| 3 | RB_SPI_IE_DMA_END | RW | DMA end interrupt enable (only supported by SPI0):<br>1: Interrupt enabled;<br>0: Interrupt disabled. | 0 |
| 2 | RB_SPI_IE_FIFO_HF | RW | More than half of FIFO used interrupt enable:<br>1: Interrupt enabled;<br>0: Interrupt disabled. | 0 |
| 1 | RB_SPI_IE_BYTE_END | RW | SPI single byte transmission completion interrupt enable:<br>1: Interrupt enabled;<br>0: Interrupt disabled. | 0 |
| 0 | RB_SPI_IE_CNT_END | RW | SPI all byte transmission completion interrupt enable:<br>1: Interrupt enabled;<br>0: Interrupt disabled. | 0 |

SPI clock divider register in master mode (R8_SPIx_CLOCK_DIV) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_SPI_CLOCK_DIV | RW | Frequency division factor in master mode, the minimum value is 2.<br>Fsck= Fsys/frequency division factor. | 10h |

SPI preset data register in slave mode (R8_SPI0_SLAVE_PRE) (only supported by SPI0)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_SPI0_SLAVE_PRE | RW | Preset data first returned in slave mode.<br>Used to receive the returned data after first byte of data. | 10h |

SPI data buffer (R8_SPIx_BUFFER) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_SPIx_BUFFER | RW | SPI data transmit and receive buffer. | XXh |

SPI working status register (R8_SPIx_RUN_FLAG) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_SPI_SLV_SELECT | RO | Chip select status in slave mode (only supported by SPI0):<br>1: Being selected;<br>0: No chip selected. | 0 |
| 6 | RB_SPI_SLV_CS_LOAD | RO | First loading status after chip select in slave mode (only supported by SPI0):<br>1: Being loading R8_SPI0_SLAVE_PRE;<br>0: Not yet loaded or has completed. | 0 |
| 5 | RB_SPI_FIFO_READY | RO | FIFO ready:<br>1: FIFO is ready (R16_SPIx_TOTAL_CNT is not 0, and the FIFO is not full when receiving or the FIFO is not empty when transmitting);<br>0: FIFO is not ready. | 0 |
| 4 | RB_SPI_SLV_CMD_ACT | RO | Command received completion status in slave mode, that is, completing the exchange of first byte data (only supported by SPI0):<br>1: That has just been exchanged is the first byte;<br>0: The first byte has not been exchanged or it is not the first byte. | 0 |
| [3:0] | Reserved | R0 | Reserved | 0000b |

SPI interrupt flag register (R8_SPIx_INT_FLAG) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_SPI_IF_FST_BYTE | RW1 | First byte received flag in slave mode (only supported by SPI0):<br>1: The first byte has been received;<br>0: The first byte is not received. | 0 |
| 6 | RB_SPI_FREE | RO | Current SPI free:<br>1: Free;<br>0: Not free. | 1 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_SPI_IF_FIFO_OV | RW1 | FIFO overflow (FIFO is full when receiving or FIFO is empty when transmitting) flag. Write 1 to reset:<br>1: Overflow;<br>0: Not overflow. | 0 |
| 3 | RB_SPI_IF_DMA_END | RW1 | DMA end flag (only supported by SPI0). Write 1 to reset:<br>1: End;<br>0: Not end. | X |

| | | | | |
|---|---|---|---|---|
| 2 | RB_SPI_IF_FIFO_HF | RW1 | More than half of FIFO used (FIFO>=4 when receiving or FIFO<4 when transmitting) flag. Write 1 to reset:<br>1: More than half of FIFO has been used;<br>0: FIFO has been used not more than half. | 0 |
| 1 | RB_SPI_IF_BYTE_END | RW1 | SPI single byte transfer end flag. Write 1 to reset:<br>1: End;<br>0: Not end. | 0 |
| 0 | RB_SPI_IF_CNT_END | RW1 | SPI all byte transfer end flag. Write 1 to reset:<br>1: All byte transfer ends;<br>0: All byte transfer not end. | 1 |

SPI transceiver FIFO count register (R8_SPIx_FIFO_COUNT) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_SPIx_FIFO_COUNT | RW | Current byte count in FIFO. | 00h |

SPI transceiver FIFO count register (R8_SPIx_FIFO_COUNT1) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7：0] | R8_SPIx_FIFO_COUNT1 | RW | Current byte count in FIFO.<br>The same as R8_SPIx_FIFO_COUNT. | 00h |

SPI transceiver data total length register (R16_SPIx_TOTAL_CNT) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_SPIx_TOTAL_CNT | RW | Total number of bytes of SPI data transceiver in master mode, and the lower 12 bits are valid. At most 4095 bytes can be received/transmitted at a time when using DMA. It is not supported in slave mode. | 0000h |

SPI data FIFO register (R8_SPIx_FIFO) (x=0/1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_SPIx_FIFO | RO/<br>WO | Data FIFO register. | XXh |

The registers R8_SPIx_BUFFER and R8_SPIx_FIFO are both SPI data related registers, and the main differences between them are:

Reading R8_SPIx_BUFFER means to obtain the data from the last exchange of SPI, and it does not affect FIFO and R8_SPIx_FIFO_COUNT,

Writing to R8_SPIx_BUFFER in master mode means to send the byte directly, and the write operation in slave mode is not defined;

Reading R8_SPIx_FIFO means to obtain the data from the earliest exchange in FIFO, which will reduce FIFO and R8_SPIx_FIFO_COUNT,

Writing to R8_SPI0_FIFO means to temporarily store the data in FIFO. In slave mode, the external SPI host decides when to take it. In master mode, the transmission is automatically started when R16_SPIx_TOTAL_CNT is not 0.

Current address of SPI0 DMA buffer (R16_SPI0_DMA_NOW)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_SPI0_DMA_NOW | RW | Current address of DMA data buffer.<br>It can be used to calculate the number of conversions.<br>COUNT=SPI0_DMA_NOW-SPI0_DM A_BEG. | XXXXh |

Start address of SPI0 DMA buffer (R16_SPI0_DMA_BEG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_SPI0_DMA_BEG | RW | Start address of DMA data buffer, only the lower 15 bits are valid. | XXXXh |

End address of SPI0 DMA buffer (R16_SPI0_DMA_END)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_SPI0_DMA_END | RW | End address of DMA data buffer (not included), only the lower 15 bits are valid. | XXXXh |

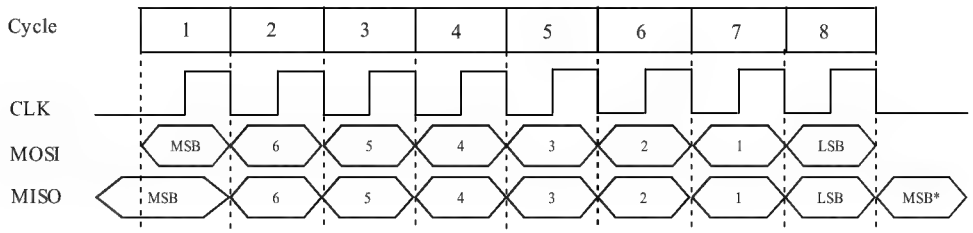
## 10.3 SPI transfer frame formats

SPI supports 2 transfer frame formats, mode0 and mode3, which can be selected by setting RB_SPI_MST_SCK_MOD in R8_SPIx_CTRL_MOD. Always sample and input serial data at rising edge of SCK, and output serial data at falling edge.

The data transmission formats are shown in the figures below:

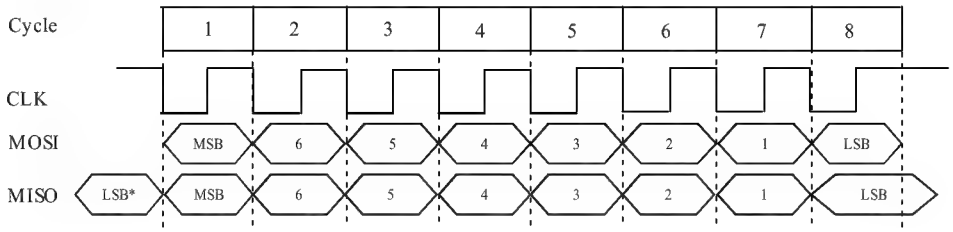
Mode0: RB_SPI_MST_SCK_MOD = 0

Figure 10-1 Transfer format of SPI mode0
模式0时序图



Mode3: RB_SPI_MST_SCK_MOD = 1

Figure 10-2 Transfer format of SPI mode3
模式3时序图




## 10.4 SPI configuration

### 10.4.1 Master mode

In SPI master mode, serial clock is generated on SCK pin, and chip select pin can be specified as any I/O pin.

Configuration procedure:

(1) Set R8_SPIx_CLOCK_DIV, to configure SPI clock frequency;

(2) Set RB_SPI_MODE_SLAVE in R8_SPIx_CTRL_MOD to 0, to configure SPI to master mode;

(3) Set RB_SPI_MST_SCK_MOD in R8_SPIx_CTRL_MOD, to select clock idle mode0 or mode3;

(4) Set the RB_SPI_FIFO_DIR in R8_SPIx_CTRL_MOD to configure the FIFO direction. If it is 1, FIFO is used to receive. If it is 0, FIFO is used to transmit;

(5) Set RB_SPI_MOSI_OE and RB_SPI_SCK_OE in R8_SPIx_CTRL_MOD to 1, and set RB_SPI_MISO_OE to 0, and set GPIO direction configuration register (R32_PA/PB_DIR), to set the MOSI pin and SCK pin as output, and MISO pin as input;

(6) SCK remains unchanged in 2-wire mode, RB_SPI_MOSI_OE=0, MOSI is not used. Input (same as 3-wire mode, RB_SPI_MISO_OE=0 and the pin is set as input) and output (RB_SPI_MISO_OE=1 and the pin is set as output) are realized by MISO half-duplex, and the directions are switched manually;

(7) Optional. If DMA is enabled, it is needed to write the start address of transceiver buffer to R16_SPI_DMA_BEG and write the end address (not included) to R16_SPI_DMA_END. It is recommended to set RB_SPI_DMA_ENABLE after setting RB_SPI_FIFO_DIR. If R16_SPIx_TOTAL_CNT is confirmed as 0, RB_SPI_DMA_ENABLE can be first set to 1 to enable DMA function.

Data transmission:

(1) Set RB_SPI_FIFO_DIR to 0, and the current FIFO direction is output;

(2) Write to the R16_SPIx_TOTAL_CNT register, and set the length of the data to be sent;

(3) Write to the R8_SPIx_FIFO register and write the data to be sent to FIFO. If R8_SPIx_FIFO_COUNT is less than FIFO capacity, continue to write FIFO. If DMA is enabled, DMA will automatically load FIFO to complete this step;

(4) As long as R16_SPIx_TOTAL_CNT is not 0 and there is data in FIFO, SPI master will automatically send data, otherwise, it will pause;

(5) Wait until R16_SPIx_TOTAL_CNT register becomes 0, indicating that the data transmission is completed. If only one byte is sent, you can also query and wait for RB_SPI_FREE to be idle or wait for R8_SPIx_FIFO_COUNT to be 0.

Data reception:

(1) Set RB_SPI_FIFO_DIR to 1, to set the current FIFO direction to input;

(2) Write to the R16_SPIx_TOTAL_CNT register, to set the length of the data to be received;

(3) As long as R16_SPIx_TOTAL_CNT is not 0 and FIFO is not full, SPI master will automatically receive data, otherwise, it will pause;

(4) Wait until R8_SPIx_FIFO_COUNT register is not 0, indicating that the return data is received, the value read in R8_SPI0_FIFO is the received data. If DMA is enabled, DMA will automatically read FIFO to complete this step.

### 10.4.2 Slave mode

SPI0 supports the slave mode. In the slave mode, SCK pin is used to receive the serial clock of SPI master connected to the external.

Configuration procedure:

(1) Set RB_SPI_MODE_SLAVE in R8_SPI0_CTRL_MOD to 1, to configure SPI0 to slave mode;

(2) Set RB_SPI_SLV_CMD_MOD in R8_SPI0_CTRL_MOD as needed, to select the slave first byte mode or data stream mode;

(3) Set RB_SPI_FIFO_DIR in R8_SPI0_CTRL_MOD, to configure the FIFO direction. If it is 1, FIFO is used to receive; if it is 0, FIFO is used to transmit;

(4) Set RB_SPI_MOSI_OE and RB_SPI_SCK_OE in R8_SPI0_CTRL_MOD to 0, and set RB_SPI_MISO_OE to 1, and set GPIO direction configuration register (R32_PA/PB_DIR) to make MOSI pin, SCK pin and SCS pin as input, MISO pin as input (support connect multiple slaves under the bus; MISO will automatically switch to output after chip select; one master with one slave is also supported) or output (only for connection of one master with one slave). In SPI slave mode, the I/O pin direction of MISO can be set as output by GPIO direction configuration register, it can also automatically switch to output during the period of valid SPI chip select. But its output data is selected by RB_SPI_MISO_OE, it outputs SPI data when it is 1, and it outputs data of GPIO data output register when it is 0. It is recommended to set the MISO pin as input, so that MISO does not output when chip select is invalid, so that SPI bus can be shared during multiple-device operation;

(5) Optional, set the preset data register (R8_SPI0_SLAVE_PRE) in SPI0 slave mode, used to be automatically loaded into the buffer for the first time after chip select for external output. After 8 clocks (that is, the first data byte is exchanged between the master and the slave), the controller will obtain the first data byte (command code) sent by the external SPI host, and the external SPI host obtains the preset data (status value) in R8_SPI0_SLAVE_PRE through exchange. The bit7 of R8_SPI0_SLAVE_PRE will be automatically loaded into the MISO pin during SCK low level period after the SPI chip select is valid. For SPI mode 0 (CLK is at low level by default), if the bit7 of R8_SPI0_SLAVE_PRE is preset, the external SPI host will obtain the preset value of bit7 of R8_SPI0_SLAVE_PRE by inquiring the MISO pin when the SPI chip select is valid but has no data transmission, thereby the value of bit7 of R8_SPI0_SLAVE_PRE can be obtained only by a valid SPI chip select (Usually a busy status is provided for the host, so that the host can quickly query);

(6) Optional. If DMA is enabled, it is needed to write the start address of transceiver buffer to R16_SPI_DMA_BEG and write the end address (not included) to R16_SPI_DMA_END. It is recommended to set RB_SPI_DMA_ENABLE after setting RB_SPI_FIFO_DIR.

Data transmission:

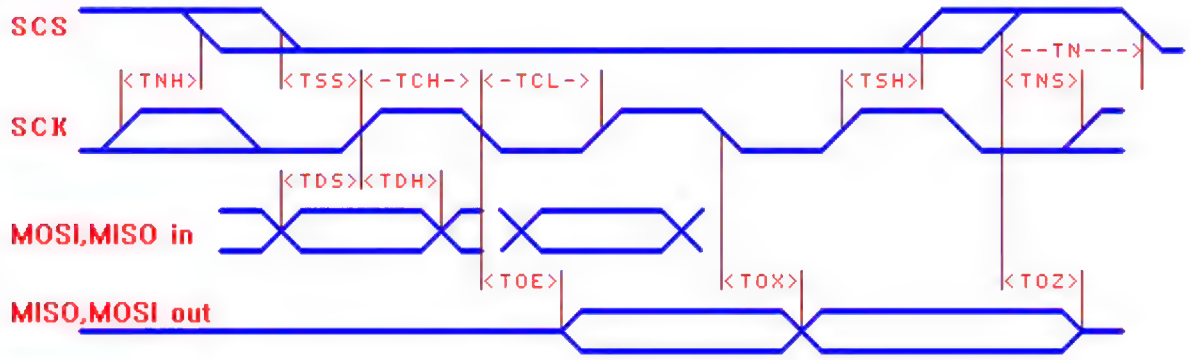
(1) Set RB_SPI_FIFO_DIR in R8_SPI0_CTRL_MOD to 0, and the current FIFO direction as output;

(2) Optional step. If DMA is enabled, it is needed to set RB_SPI_DMA_ENABLE to 1 to enable DMA function;

(3) Write multiple transmission data into FIFO register R8_SPI0_FIFO, and the external host determines when to take it away. If DMA is enabled, DMA will automatically load FIFO to complete this step;
(4) Query R8_SPI0_FIFO_COUNT, if it is not full, continue to write data to be sent to FIFO.

Data reception:
(1) Set RB_SPI_FIFO_DIR of R8_SPI0_CTRL_MOD to 1, and the current FIFO direction as input;
(2) Optional step. If DMA is enabled, it is needed to set RB_SPI_DMA_ENABLE to 1 to enable DMA function;
(3) Query R8_SPI0_FIFO_COUNT, if it is not empty, the data has been received and the data will be taken away by reading R8_SPI0_FIFO. If DMA is enabled, DMA will automatically read FIFO to complete this step;
(4) For reception of the single byte data, R8_SPI0_BUFFER can be read directly without using FIFO.

## 10.5 SPI Timing




| Symbol | Parameter description (TA=25°C, VIO33=3.3V) | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| TSS | Setup time of valid SCS before SCK rising edge | Tsys*1.05 | | | nS |
| TSH | Hold time of valid SCS before SCK rising edge | Tsys*1.05 | | | nS |
| TNS | Setup time of invalid SCS before SCK rising edge | 15 | | | nS |
| TNH | Hold time of invalid SCS before SCK rising edge | 15 | | | nS |
| TN | Time of invalid SCS (interval time of SPI operation) | Tsys*2 | | | nS |
| TCH | Time of SCK clock at high level | Tsys*0.55 | | | nS |
| TCL | Time of SCK clock at low level | Tsys*0.55 | | | nS |
| TDS | Setup time of MOSI/MISO input before SCK rising edge | 8 | | | nS |
| TDH | Hold time of MOSI/MISO input before SCK rising edge | 5 | | | nS |
| TOE | SCK falling edge to MISO/MOSI output valid | 0 | | 18 | nS |
| TOX | SCK falling edge to MISO/MOSI output change | 0 | 5 | 16 | nS |
| TOZ | SCS invalid to MISO/MOSI output invalid | 2 | | 24 | nS |

*Note: Tsys is the cycle of system clock frequency (1/Fsys).*

# Chapter 11 PWM

## 11.1 Introduction to PWM

In addition to the 4-channel 26-bit PWM output provided by timer, the system also provides 8-channel 8-bit PWM output (PWM4~PWM11), with adjustable duty ratio and 8 optional cycles as the fixed PWM cycle, so the operation is simple.

## 11.2 Register description

Table 11-1 PWMx registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_PWM_OUT_EN | 0x40005000 | PWMx output enable register | 0x00 |
| R8_PWM_POLAR | 0x40005001 | PWMx output polarity configuration register | 0x00 |
| R8_PWM_CONFIG | 0x40005002 | PWMx configuration control register | 0x0X |
| R8_PWM_CLOCK_DIV | 0x40005003 | PWMx clock divider register | 0x00 |
| R32_PWM4_7_DATA | 0x40005004 | PWM4/5/6/7 data hold register | 0xXXXXXXXX |
| R8_PWM4_DATA | 0x40005004 | PWM4 data hold register | 0xXX |
| R8_PWM5_DATA | 0x40005005 | PWM5 data hold register | 0xXX |
| R8_PWM6_DATA | 0x40005006 | PWM6 data hold register | 0xXX |
| R8_PWM7_DATA | 0x40005007 | PWM7 data hold register | 0xXX |
| R32_PWM8_11_DATA | 0x40005008 | PWM8/9/10/11 data hold register | 0xXXXXXXXX |
| R8_PWM8_DATA | 0x40005008 | PWM8 data hold register | 0xXX |
| R8_PWM9_DATA | 0x40005009 | PWM9 data hold register | 0xXX |
| R8_PWM10_DATA | 0x4000500A | PWM10 data hold register | 0xXX |
| R8_PWM11_DATA | 0x4000500B | PWM11 data hold register | 0xXX |
| R8_PWM_INT_CTRL | 0x4000500C | PWMx interrupt control and status register | 0x00 |

PWMx output enable register (R8_PWM_OUT_EN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_PWM11_OUT_EN | RW | PWM11 output enable:<br>1: Enabled; 0: Disabled. | 0 |
| 6 | RB_PWM10_OUT_EN | RW | PWM10 output enable:<br>1: Enabled; 0: Disabled. | 0 |
| 5 | RB_PWM9_OUT_EN | RW | PWM9 output enable:<br>1: Enabled; 0: Disabled. | 0 |
| 4 | RB_PWM8_OUT_EN | RW | PWM8 output enable:<br>1: Enabled; 0: Disabled. | 0 |
| 3 | RB_PWM7_OUT_EN | RW | PWM7 output enable:<br>1: Enabled; 0: Disabled. | 0 |

| | | | | |
|---|---|---|---|---|
| 2 | RB_PWM6_OUT_EN | RW | PWM6 output enable:<br>1: Enabled; 0: Disabled. | 0 |
| 1 | RB_PWM5_OUT_EN | RW | PWM5 output enable:<br>1: Enabled; 0: Disabled. | 0 |
| 0 | RB_PWM4_OUT_EN | RW | PWM4 output enable:<br>1: Enabled; 0: Disabled. | 0 |

PWMx output polarity configuration register (R8_PWM_POLAR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_PWM11_POLAR | RW | PWM11 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 6 | RB_PWM10_POLAR | RW | PWM10 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 5 | RB_PWM9_POLAR | RW | PWM9 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 4 | RB_PWM8_POLAR | RW | PWM8 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 3 | RB_PWM7_POLAR | RW | PWM7 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 2 | RB_PWM6_POLAR | RW | PWM6 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 1 | RB_PWM5_POLAR | RW | PWM5 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |
| 0 | RB_PWM4_POLAR | RW | PWM4 output polarity control:<br>1: Default at high level, active low;<br>0: Default at low level, active high. | 0 |

PWMx configuration control register (R8_PWM_CONFIG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_PWM10_11_STAG_EN | RW | PWM10/11 interleaved output enable:<br>1: Interleaved output;<br>0: Independent output. | 0 |
| 6 | RB_PWM8_9_STAG_EN | RW | PWM8/9 interleaved output enable:<br>1: Interleaved output;<br>0: Independent output. | 0 |
| 5 | RB_PWM6_7_STAG_EN | RW | PWM6/7 interleaved output enable:<br>1: Interleaved output; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Independent output. | |
| 4 | RB_PWM4_5_STAG_EN | RW | PWM6/7 interleaved output enable:<br>1: Interleaved output;<br>0: Independent output. | 0 |
| [3:2] | RB_PWM_CYC_MOD | RW | PWM data width selection:<br>00: 8-bit data width; 01: 7-bit data width;<br>00: 6-bit data width; 11: 5-bit data width. | 00b |
| 1 | RB_PWM_STAG_ST | RO | PWM interleave flag:<br>1: PWM5/7/9/11 is allowed to output;<br>0: PWM4/6/8/10 is allowed to output. | X |
| 0 | RB_PWM_CYCLE_SEL | RW | PWM cycle selection, matching with PWM data width:<br>1: 255/127/63/31 clock cycles;<br>0: 256/128/64/32 clock cycles. | 0 |

PWMx clock divider register (R8_PWM_CLOCK_DIV)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_PWM_CLOCK_DIV | RW | PWM reference clock frequency division factor.<br>Fpwm=Fsys/R8_PWM_CLOCK_DIV. | 00h |

PWM data hold register group1 (R32_PWM4_7_DATA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | R8_PWM7_DATA | RW | PWM7 data hold register. | XXb |
| [23:16] | R8_PWM6_DATA | RW | PWM6 data hold register. | XXb |
| [15:8] | R8_PWM5_DATA | RW | PWM5 data hold register. | XXb |
| [7:0] | R8_PWM4_DATA | RW | PWM4 data hold register. | XXb |

PWM data hold register group2 (R32_PWM8_11_DATA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [31:24] | R8_PWM11_DATA | RW | PWM11 data hold register. | XXb |
| [23:16] | R8_PWM10_DATA | RW | PWM10 data hold register. | XXb |
| [15:8] | R8_PWM9_DATA | RW | PWM9 data hold register. | XXb |
| [7:0] | R8_PWM8_DATA | RW | PWM8 data hold register. | XXb |

PWMx interrupt control and status register (R8_PWM_INT_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_PWM_IF_CYC | RW1 | PWM cycle end flag. Write 1 to reset:<br>1: End; 0: Not end. | 0 |
| [6:2] | Reserved | RO | Reserved | 00000b |
| 1 | RB_PWM_CYC_PRE | RW | Select the interrupt time point at the end of PWM cycle:<br>1: Interrupt is generated 16 counts in advance (take | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 8-bit data width as an example, interrupt is generated when the count reaches 240);<br>0: Interrupt is generated 2 counts in advance (take 8-bit data width as an example, interrupt is generated when the count reaches 254). | |
| 0 | RB_PWM_IE_CYC | RW | PWM cycle end interrupt enable:<br>1: Enabled; 0: Disabled. | 0 |

## 11.3 PWM configuration

(1) Set the R8_PWM_CLOCK_DIV register, to configure the reference clock frequency of PWM;
(2) Set the PWM output polarity configuration register (R8_PWM_POLAR), to configure the output polarity of the corresponding PWMx;
(3) Set the PWM configuration control register (R8_PWM_CONFIG), to set the PWM mode, data width and cycle;
(4) Set PWM output enable register (R8_PWM_OUT_EN), to enable the corresponding PWMx output;
(5) Calculate the data according to the required duty cycle and write it into the corresponding data hold register (R8_PWMx_DATA);
(6) Set the required PWM pin direction of PWM4-PWM11 as output, optional, set the drive capability of corresponding I/O;
(7) Update the data in R8_PWMx_DATA as needed, to update the output duty cycle.

The duty cycle of output PWM can be modified by adjusting R8_PWMx_DATA register. The calculation formula of duty cycle is:

Ncyc = RB_PWM_CYCLE_SEL ? (2^n-1) : (2^n)
(n= data bit width), the Ncyc result ranges from 63 to 256.
Duty cycle of PWMx output valid level = R8_PWMx_DATA / Ncyc * 100%
PWMx output frequency Fpwmout = Fpwm / Ncyc = Fsys/R8_PWM_CLOCK_DIV / Ncyc

To generate a DC signal through PWM, you can use R/C and other circuits to filter at PWMx output end. It is recommended to use a 2-stage RC with a time constant much larger than 4 / Fpwmout, or a 1-stage RC with a time constant much larger than 100 / Fpwmout.

# Chapter 12 Inter Integrated Circuit (I2C) Interface

## 12.1 Introduction to I2C

I2C (inter-integrated circuit) bus interface is a medium-low-speed serial bus. Multiple masters and slaves can be connected on the bus. Usually, I2C interface consists of 2 pins: serial clock pin (SCL) and serial data pin (SDA).

### 12.1.1 Main features

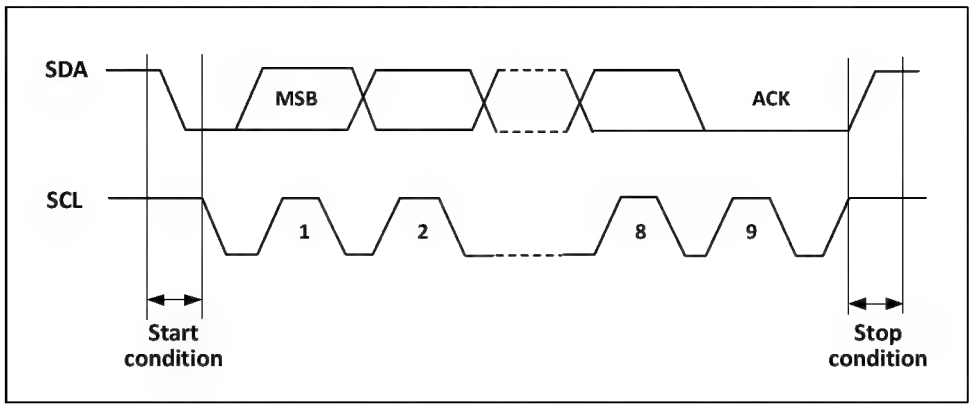
CH583 and CH582 both provide I2C interface.

- Master mode and Slave mode, support multi-master and multi-slave.
- Two speeds: 100KHz and 400KHz, compatible with I2C Bus Specification.
- 7-bit and 10-bit addressing.
- The slave device supports dual 7-bit addresses.
- Broadcast bus.
- Bus arbitration, error detection, PEC verification, clock extension.
- SMBus compatibility.

## 12.2 I2C overview

I2C is a half-duplex bus and can operate in one of the 4 following modes: master transmitter, master receiver, slave transmitter and slave receiver. By default, I2C operates in slave mode. After I2C generates a ATART condition, it automatically switches from slave to master. After an arbitration is lost or a Stop signal generates, it switches to slave. I2C supports multi-master mode. When it operates in master mode, I2C transmits data and addresses actively. Data and addresses are transferred as 8-bit bytes, high bits first, low bits last. After a Start event, it is 1-byte (in 7-bit mode) or 2-byte (in 10-bit mode). Every time the master transmits 8-bit data or address, the slave needs to respond one ACK, that is, pull down SDA bus, as shown in Figure 12-1.

Figure 12-1 I2C timing diagram




## 12.3 Master mode

In master mode, I2C dominates a data transmission and generates the clock signal. A data transmission always begins with a Start condition, and ends with a Stop condition. The following is the step sequence in master mode:

1) Set the correct timings in the control register2 (R16_I2C_CTRL2) and clock control register (R16_I2C_CKCFGR);

2) Set suitable rise time in rise time register (R16_I2C_RTR);
3) Set the PE bit to enable peripheral in the control register (R16_I2C_CTRL1);
4) Set the START bit in the control register (R16_I2C_CTRL1), to generate a Start condition.
   After the START bit is set, I2C automatically switch to master mode, MSL bit is set, a Start condition 产 is generated. After a Start condition is generated, SB bit will be set. If ITEVTEN bit (in R16_I2C_CTRL2) is set, an interrupt will be generated. At this time, it is required to read the status register1 (R16_I2C_STAR1), write slave address to the data register, SB bit will be automatically cleared;
5) If in 10-bit addressing mode, write the data register to send the header sequence (header sequence is 11110xx0b, where xx denotes the highest 2 bits of 10-bit address).

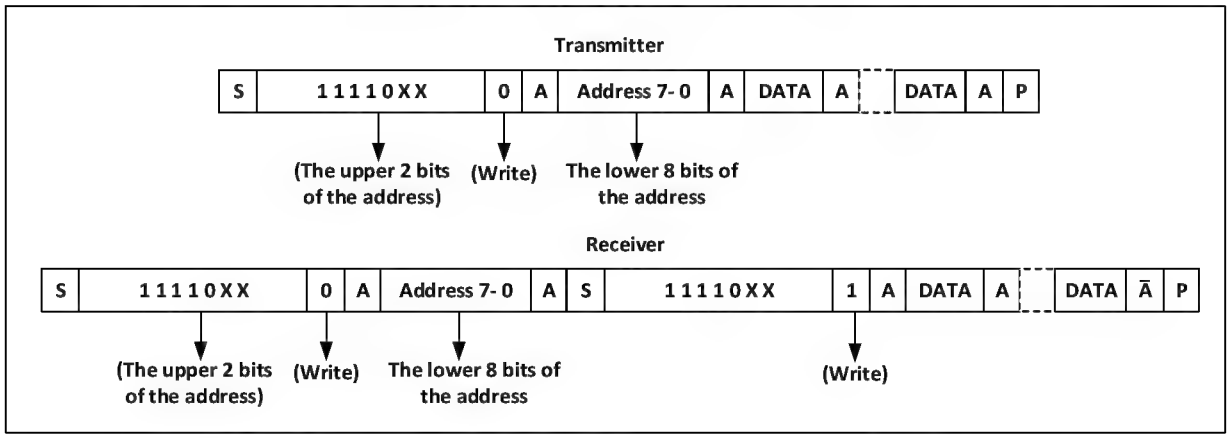
   After the header sequence is sent, the ADD10 bit of the status register will be set. If ITEVTEN bit is set, an interrupt will be generated. At this time, it is required to read R16_I2C_STAR1 register, then write the second address byte to the data register, to clear ADD10 bit.

   Then, write the data register to send the second address byte. After the second address byte is sent, ADDR bit of the status register will be set. If ITEVTEN bit is set, an interrupt will be generated. At this time, it is required to read R16_I2C_STAR1 register and then read R16_I2C_STAR2 register to clear ADDR bit;

   If in 7-bit addressing mode, write data register to send address byte. After the address byte is sent, ADDR bit of the status register will be set. If ITEVTEN bit is set, an interrupt will be generated. At this time, it is required to read R16_I2C_STAR1 register and then read R16_I2C_STAR2 register to clear ADDR bit; In 7-bit addressing mode, the first sent byte is the address byte. The first 7 bits represent target slave device address, and the 8th bit determines the direction of the following message. 0 means that the master writes data to the slave, and 1 means that the master reads data from the slave.

   In 10-bit addressing mode, as shown in Figure 12-3, when transmitting address, the first byte is 11110xx0, where xx denotes the highest 2 bits of the 10-bit address. The second byte is the lower 8 bits of the 10-bit address. If going into master transmitter mode later, continue to transmit data. If going into master receiver mode later, it is required to send a repeated Start condition, followed by the header (11110xx1), then enter master receiver mode.

Figure 12-3 Master transceiver data in 10-bit mode




6) For transmitter, the internal shift register of the master transmits data from the data register to SDA. When the master receives ACK, the TxE bit of the status register1 (R16_I2C_STAR1) is set. If ITEVTEN bit and ITBUFEN bit are set, an interrupt will be generated. Write data to the data register to clear TxE bit.

If TxE bit is set and no data is written to the data register before last data transmission, BTF bit will be set. SCL is kept at low level before it is cleared. Read R16_I2C_STAR1, and then write data into the data register, to clear BTF bit.

For receiver, I2C receives data from SDA, and writes data to the data register via the shift register. After each byte, if ACK bit is set, I2C will send a response low level, and RxNE bit will be set. If ITEVTEN bit and ITBUFEN bit are set, an interrupt will be generated. If RxNE is set and no original data is read out before new data reception, BTF bit will be set. Before BTF is cleared, SCL will be kept at low level. Read R16_I2C_STAR1, then read the data register, to clear BTF bit.

7) The master automatically sends a Stop condition when it ends sending data, that is, STOP bit is set. For receiver, the master is required to set NAK at the response bit of the last data bit. Note, after NAK is generated, I2C will automatically switch to slave mode.

## 12.4 Slave mode

In slave mode, I2C can recognize its address and general call address. The recognition of general call address can be enabled or disabled by software. Once a Start condition is detected, I2C will compare the data of SDA with its own address via the shift register (bit number is determined by ENDUAL and ADDMODE) or general address (ENGC bit is set). If not matched, the interface ignores it and waits for another Start condition. If the header is matched, the ACK signal generates and waits for the address of the second byte. If the address of the second byte is matched or the full address is matched in 7-bit addressing mode, the ACK is generated firstly; the ADDR bit is set, if ITEVTEN bit has been set, the corresponding interrupt will be generated. If in dual-address mode (ENDUAL bit is set), it is required to read DUALF bit to judge which address is woken up by the master.

For slave mode, it is receiver mode by default. When the last bit of the received header is 1, or when the last bit of the 7-bit address is 1 (the first received is the header or normal 7-bit address), I2C will go into transmitter mode, TRA bit indicates that it is in receiver mode or in transmitter mode.

In transmitter mode, after ADDR bit is cleared, I2C transmits byte from the data register to SDA via the shift register. After the ACK is received, TxE bit will be set. If ITEVTEN bit and ITBUFEN bit are set, an interrupt will be generated. If TxE is set but no new data is written to the data register before the end of next data transmission, BTF bit will be set. Before BTF is cleared, SCL will be kept at low level. Read the status register1 (R16_I2C_STAR1), then write data to the data register, to clear BTF bit.

In receiver mode, after ADDR is cleared, I2C stores data in SDA to the data register via the shift register. Every time a byte is received, I2C will set ACK bit, and set RxNE bit. If ITEVTEN and ITBUFEN are set, an interrupt will be generated. If RxNE is set, and no previous data is read out before reception of new data, BTF bit will be set. Before BTF bit is cleared, SCL will be kept at low level. Read the status register1 (R16_I2C_STAR1), and read the data in the data register, to clear BTF bit.

When I2C detects a Stop condition, STOPF bit will be set. If ITEVFEN bit is set, an interrupt will be generated. Read the status register (R16_I2C_STAR1), then write to the control register (such as reset control word SWRST) to clear.

## 12.5 Error conditions

### 12.5.1 Bus error (BERR)

The bus error occurs when I2C interface detects an external Stop or Start condition during an address or a data transfer. When a bus error occurs, the BERR bit is set. And an interrupt is generated if the ITERREN is set. In slave mode, data are discarded, and the lines are released by hardware. If it is a Start condition, the slave considers it is a restart, and waits for an address or a Stop condition. If it is a Stop condition, the slave behaves like for a Stop condition. In master mode, lines are not released by hardware and the state of the current transmission is not affected. The user codes determine to abort or not the current transmission.

### 12.5.2 Acknowledge failure (AF)

When I2C interface detects a non-acknowledge bit, the acknowledge failure occurs. When it occurs, the AF bit is set, and an interrupt is generated if the ITERREN bit is set. When AF occurs, if I2C interface works in slave mode, the lines must be released by hardware; if in master mode, a Stop condition must be generated by software.

### 12.5.3 Arbitration lost (ARLO)

When I2C interface detects an arbitration lost condition, it occurs. When an arbitration lost occurs, the ARLO bit is set, and an interrupt is generated if the ITERREN bit is set. I2C switches to slave mode, and does not acknowledge its slave address in the same transfer, unless the master initiates a new Start condition. Lines are released by hardware.

### 12.5.4 Overload/underload error (OVR)

1) Overrun error:

In slave mode, when clock stretching is disabled and I2C interface is receiving data, if a byte is received and the last received data has not been read, it occurs. When it occurs, the last received byte is lost, and the transmitter should re-transmit the last received byte.

2) Underrun error:

In slave mode, when clock stretching is disabled and I2C interface is transmitting data, if no new data is written to the data register before the next byte, it occurs. When it occurs, the data in the last data register will be sent again. If it occurs, the receiver should discard the repeated received data. In order to not generate underrun error, I2C interface should write the data to the data register before the first rising edge of the next byte.

## 12.6 Clock stretching

If clock stretching is disabled, overrun/underrun error may occur. However, if clock stretching is enabled:

1) In transmitter mode, if the TxE bit is set and the BTF bit is set, SCL is kept at low level and waits for user to read the status register, and write the data to be sent into the data register;
2) In receiver mode, if the RxNE bit is set and the BTF bit is set, SCL is kept at low level after data is received, until user reads the status register and read the data register;

So, enabling clock stretching helps avoid overrun/underrun error.

## 12.7 SMBus

The System Management Bus (SMBus) is a 2-wire interface, which is usually used between system and power management. SMBus has multiple similarities with I2C, for example, SMBus uses the same 7-bit addressing mode as I2C. The following is similarities between SMBus and I2C:

1) Master-slave communication, Master provides clock, multi-master multi-slave is supported;
2) Two-wire communication protocol;
3) 7-bit addressing format.

Differences:

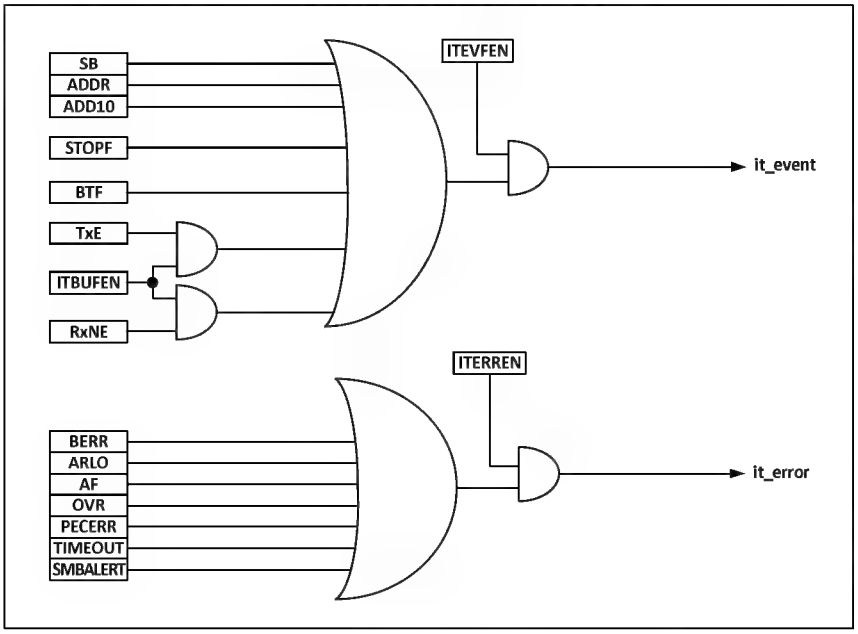
1) The speed of I2C can be up to 400KHz. The speed of SMBus can only be up to 100KHz, and the minimum speed of SMBus is 10KHz;
2) SMBus: 35 mS clock low timeout. I2C: no timeout;
3) SMBus has fixed logic levels. While I2C does not have fixed logic levels, which are determined by VDD;
4) SMBus has bus protocols, while I2C does not have bus protocols.

SMBus also contains device identification, address resolution protocol, unique device identifier, SMBus alert and various bus protocols, please refer to SMBus specification version 2.0 for details. When SMBus is used, set the SMBus bit of the control register, and configure the SMBTYPE bit and the ENAARP bit as required.

## 12.8 Interrupt

I2C interface provides event interrupt and error interrupt. After going into the same interrupt service program, they are processed separately after query.

Figure 12-4 I2C interrupt requests




## 12.9 Interrupt

A PEC calculator is a CRC8 calculator, which has been implemented to improve transmission reliability, by using the following polynomial serially on each bit: $C=X^8+X^2+X+1$.

PEC calculation is enabled by the ENPEC bit of the control register, and calculated on all message bytes, including addressed and Read/Write bits. In transmission, if PEC is enabled, the CRC8 calculation result of the last byte is added after the last byte data. In reception, the last byte is considered as the CRC8 calculation result. If it is not

matched with the internal calculation result, a NAK is sent. In case of master receiver, a NAK follows the PEC whatever the check result.

## 12.10 Register description

Table 12-1 I2C registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R16_I2C_CTRL1 | 0x40004800 | I2C control register 1 | 0x0000 |
| R16_I2C_CTRL2 | 0x40004804 | I2C control register 2 | 0x0000 |
| R16_I2C_OADDR1 | 0x40004808 | I2C address register 1 | 0x0000 |
| R16_I2C_OADDR2 | 0x4000480C | I2C address register 2 | 0x0000 |
| R16_I2C_DATAR | 0x40004810 | I2C data register | 0x0000 |
| R16_I2C_STAR1 | 0x40004814 | I2C status register 1 | 0x0000 |
| R16_I2C_STAR2 | 0x40004818 | I2C status register 2 | 0x0000 |
| R16_I2C_CKCFGR | 0x4000481C | I2C clock register | 0x0000 |
| R16_I2C_RTR | 0x40004820 | I2C rise time register | 0x0002 |

I2C control register (I2C_CTRL1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | SWRST | RW | Software reset. When user codes set this bit, I2C is reset. Before reset, make sure that I2C lines are released and the bus is idle.<br>*Note: This bit can reset I2C when no Stop condition is detected on the bus but the busy bit is 1.* | 0 |
| 14 | Reserved | RO | Reserved | 0 |
| 13 | ALERT | RW | SMBus alert. This bit can be set or cleared by user codes. After the PE bit is set, this bit can be cleared by hardware.<br>1: Drive SMBusALERT pin low, response address header followed by ACK;<br>0: Release SMBusALERT pin high, response address header followed by NACK. | 0 |
| 12 | PEC | RW | Data packet error checking enable, it is set to enable data packet error checking detection. This bit can be set or cleared by user codes. After PEC is transmitted, or when a Start or Stop condition is generated, or when the PE bit is cleared to 0, this bit is cleared by hardware.<br>1: PEC;<br>0: No PEC.<br>*Note: PEC is corrupted when an arbitration is lost.* | 0 |
| 11 | POS | RW | ACK/PEC position. This bit can be set or cleared by user codes. Or this bit can be cleared by hardware | 0 |

| | | | | |
|---|---|---|---|---|
| | | | after PE is cleared.<br>1: The ACK bit controls ACK/NAK of the next byte that is received in the shift register. The next byte that is received in PEC shift register is a PEC;<br>0: The ACK bit controls ACK/NAK of the current byte being received in the shift register. The PEC bits indicates that current byte in the shift register is a PEC.<br>*Note: The POS bit is used for the reception of 2bytes data: it must be configured before reception. To NACK the second byte, the ACK bit must be cleared at once after the ADDR bit is cleared. To check the second byte as PEC, the PEC bit must be set after ADDR event occurs and the POS bit is configured.* | |
| 10 | ACK | RW | Acknowledge enable. This bit can be set or cleared by user codes. Or this bit can be cleared by hardware after PE is set.<br>1: Acknowledge returned after a byte is received;<br>0: No acknowledge. | 0 |
| 9 | STOP | RW | Stop condition generation. This bit can be set or cleared by user codes. Or this bit can be cleared by hardware when a Stop condition is detected. Or this bit can be set by hardware when a timeout error is detected.<br>In master mode:<br>1: A Stop condition is generated during current byte transmission or after current Start condition is sent;<br>0: No Stop condition generated. In slave mode:<br>1: Release SCL and SDA after current byte transmission;<br>0: No Stop condition generated. | 0 |
| 8 | START | RW | Start condition generation. This bit can be set or cleared by user codes. Or this bit can be cleared by hardware when a Start condition is sent or PE is cleared.<br>In master mode:<br>1: Repeated Start condition generation;<br>0: No Start condition generation.<br>In slave mode:<br>1: A Start condition is generated when the bus is idle;<br>0: No Start condition generated. | 0 |
| 7 | NOSTRETCH | RW | Clock stretching disable. This bit can be used to disable clock stretching when ADDB or BTF flag is set, until it is cleared by software.<br>1: Disable clock stretching; | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 0: Enable clock stretching. | |
| 6 | ENGC | RW | General call enable. This can be set to enable general call. Response general address 00h. | 0 |
| 5 | ENPEC | RW | PEC enable. This bit can be set to enable PEC calculation. | 0 |
| 4 | ENARP | RW | ARP enable. This bit can be set to enable ARP.<br>If SMBTYPE=0, use default SMBus device address;<br>If SMBTYPE=1, use SMBus host address. | 0 |
| 3 | SMBTYPE | RW | SMBus device type.<br>1: SMBus host;<br>0: SMBus device. | 0 |
| 2 | Reserved | RO | Reserved | 0 |
| 1 | SMBUS | RW | SMBus mode selection:<br>1: SMBus mode;<br>0: I2C mode. | 0 |
| 0 | PE | RW | I2C peripheral enable.<br>1: I2C enabled; 0: I2C disabled. | 0 |

I2C control register2 (I2C_CTRL2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:11] | Reserved | RO | Reserved. | 0 |
| 10 | ITBUFEN | RW | Buffer interrupt enable.<br>1: When TxE or RxEN is set, event interrupt is generated;<br>0: When TxE or RxEN is set, no interrupt is generated. | 0 |
| 9 | ITEVTEN | RW | Event interrupt enable. This bit can be set to enable event interrupt.<br>The interrupt is generated when:<br>SB=1 (Master);<br>ADDR=1 (Master/Slave);<br>ADDR10=1 (Master);<br>STOPF=1 (Slave);<br>BTF=1, with no TxE or RxEN event;<br>If ITBUFEN=1, TxE event to 1;<br>If ITBUFEN=1, RxNE event to 1. | 0 |
| 8 | ITERREN | RW | Error interrupt enable. This bit can be set to enable error interrupt.<br>The interrupt is generated when:<br>BERR=1; ARLO=1; AF=1; OVR=1; PECERR=1; TIMEOUT=1; SMBAlert=1. | 0 |
| [7:6] | Reserved | RO | Reserved. | 0 |

| | | | | |
|---|---|---|---|---|
| [5:0] | FREQ | RW | I2C clock frequency. The correct clock frequency must be input to generate correct timing. The allowed range is 2~36MHz. These bits must be set between 000010b and 100100b, and the unit is MHz.<br>Recommended: Minimum 2MHz input clock in standard mode; minimum 4MHz input clock in fast mode. | 0 |

I2C address register1 (I2C_OADDR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | ADDMODE | RW | Addressing mode.<br>1: 10-bit slave address (7-bit address not acknowledged);<br>0: 7-bit slave address (10-bit address not acknowledged). | 0 |
| 14 | MUST1 | RW1 | Must be kept at 1 by software. | 0 |
| [13:10] | Reserved | RO | Reserved | 0 |
| [9:8] | ADD9_8 | RW | Bus address. Bits9:8 when using 10-bit address.<br>Ignore when using 7-bit address. | 0 |
| [7:1] | ADD7_1 | RW | Bus address, bits7:1. | 0 |
| 0 | ADD0 | RW | Bus address. Bit0 when using 10-bit address.<br>Ignore when using 7-bit address. | 0 |

I2C address register2 (I2C_OADDR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| [7:1] | ADD2 | RW | Bus address, bits7:1 in dual addressing mode. | 0 |
| 0 | ENDUAL | RW | Dual addressing mode enable bit. This bit can be set to make ADD2 recognized. | 0 |

I2C data register (I2C_DATAR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | Reserved | RO | Reserved. | 0 |
| [7:0] | DATAR | RW | Data register. It can be used to store the received data or store the data to be sent to the bus. | 0 |

I2C status register1 (I2C_STAR1)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 15 | SMBALERT | RW0 | SMBus alert. This bit can be reset by user codes writing 0, or reset by hardware when PE is at low level.<br>In SMBus host mode:<br>1: SMBus alert occurs;<br>0: No SMBus alert.<br>In SMBus slave mode:<br>1: SMBAlert response address header to SMBAlert LOW received;<br>0: No SMBAlert response address header received. | 0 |
| 14 | TIMEOUT | RW0 | Timeout or Tlow error flag. This bit can be reset by user codes writing 0, or reset by hardware when PE is at low level.<br>1: SCL remained LOW for 25mS, or master low extend time more than 10mS, or slave low extend time more than 25mS;<br>0: No timeout error.<br>*Note: When this bit is set in slave mode, slave resets communication, lines are released by hardware. When this bit is set in host mode, a Stop condition is sent by hardware.* | 0 |
| 13 | Reserved | RO | Reserved. | 0 |
| 12 | PECERR | RW0 | PEC error flag in reception. This bit can be reset by user codes writing 0, or reset by hardware when PE is at low level.<br>1: PEC error, NAK returned after PEC is received;<br>0: No PEC error. | 0 |
| 11 | OVR | RW0 | Overrun/Underrun flag.<br>1: Overrun/Underrun: If NOSTRETCH=1, when a new byte is received in reception and data register has not been read, the new received byte is lost. In transmission, when no new data is written into data register, the same byte is sent twice;<br>0: No overrun/underrun. | 0 |
| 10 | AF | RW0 | Acknowledge failure flag. This bit can be reset by user codes writing 0, or reset by hardware when PE is at low level.<br>1: Acknowledge failure;<br>0: No acknowledge failure. | 0 |

| | | | | |
|---|---|---|---|---|
| 9 | ARLO | RW0 | Arbitration lost flag. This bit can be reset by user codes writing 0, or reset by hardware when PE is at low level.<br>1: Arbitration lost detected, the interface loses the control to the bus;<br>0: No arbitration lost detected. | 0 |
| 8 | BERR | RW0 | Bus error flag. This bit can be reset by user codes writing 0, or reset by hardware when PE is at low level.<br>1: Start/Stop condition error;<br>0：No Start/Stop condition error. | 0 |
| 7 | TxE | RO | Data register empty flag. It can be cleared by writing data to the data register, or it can be cleared by hardware after a Start/Stop condition is generated or when PE is 0.<br>1: Data register empty in transmission;<br>0: Data register not empty. | 0 |
| 6 | RxNE | RO | Data register not empty flag. It can be cleared by reading/writing data to the data register, or it can be cleared by hardware when PE is 0.<br>1: Data register not empty in reception;<br>0: Data register empty. | 0 |
| 5 | Reserved | RO | Reserved. | 0 |
| 4 | STOPF | RO | Stop condition flag. It can be cleared by write operation to the control register1 after user reads the status register1, or it can be cleared by hardware when PE is 0.<br>1: Slave detects a Stop condition on the bus after an acknowledge;<br>0: No Stop condition detected. | 0 |
| 3 | ADD10 | RO | 10-bit address header sent flag. It can be cleared by write operation to the control register1 after user reads the status register1, or it can be cleared by hardware when PE is 0.<br>1: In 10-bit addressing mode, slave has sent the first address byte;<br>0: None. | 0 |

| | | | | |
|---|---|---|---|---|
| 2 | BTF | RO | Byte transmission finished flag. It can be cleared by read/write operation to the data register after user reads the status register1. In transmission, it can be cleared by hardware after a Start/Stop condition is generated or when PE is 0.<br>1: Byte transmission finished. If NOSTRETCH=0: in transmission, when a new data is sent and no new data is written into the data register; in reception, when a new byte is received but the data register has not been read;<br>0: None. | 0 |
| 1 | ADDR | RW0 | Address sent/matched flag. It can be cleared by read operation to the status register2 after user reads the status register1, or it can be cleared by hardware when PE is 0.<br>In master mode:<br>1: Address transmission finished: in 10-bit addressing mode, this bit is set after ACK of the second address byte is received; in 7-bit addressing mode, this bit is set after ACK of the byte is received;<br>0: Address transmission not finished.<br>In slave mode:<br>1: Received address matched;<br>0: Address not matched or no address received. | 0 |
| 0 | SB | RO | Start bit sent flag. It can be cleared by write operation to the data register after user reads the status register1, or it can be cleared by hardware when PE is 0.<br>1: Start bit sent; 0: Start bit not sent. | 0 |

I2C status register2 (I2C_STAR2)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:8] | PEC | RO | Packet error checking register. When PEC is enabled (ENPEC is set), this register stores the value of PEC. | 0 |
| 7 | DUALF | RO | Dual flag. It can be cleared by hardware when a Stop/Start bit is generated or when PE=0.<br>1: Received address matched with OADDR2;<br>0: Received address matched with OADDR1. | 0 |

| | | | | |
|---|---|---|---|---|
| 6 | SMBHOST | RO | SMBus host header flag. It can be cleared by hardware when a Stop/Start bit is generated or when PE=0.<br>1: When SMBTYPE=1 and ENARP=1, SMBus host address received;<br>0: SMBus host address not received. | 0 |
| 5 | SMBDEFAULT | RO | SMBus device default address flag. It can be cleared by hardware when a Stop/Start bit is generated or when PE=0.<br>1: When ENARP=1, SMBus device default address received;<br>0: SMBus device default address not received. | 0 |
| 4 | GENCALL | RO | General call address flag. It can be cleared by hardware when a Stop/Start bit is generated or when PE=0.<br>1: When ENGC=1, general call address received;<br>0: General call address not received. | 0 |
| 3 | Reserved | RO | Reserved. | 0 |
| 2 | TRA | RO | Transmission/reception flag. It can be cleared by hardware when a Stop condition is detected (STOPF=1) or repeated Start condition is detected or an arbitration lost is detected (ARLO=1) or when PE=0.<br>1: Data transmitted;<br>0: Data received.<br>This bit is determined by the R/W bit of the address byte. | 0 |
| 1 | BUSY | RO | Bus busy flag. It can be cleared when a Stop condition is detected. When the interface is disabled (PE=0), the information is still updated.<br>1: Busy bus: SDA or SCL LOW;<br>0: Idle bus, and no communication. | 0 |
| 0 | MSL | RO | Master/slave mode indication. It can be cleared by hardware when the interface is in Master mod (SB=1). It can be cleared by hardware when the bus detects a Stop bit or an arbitration lost or when PE=0. | 0 |

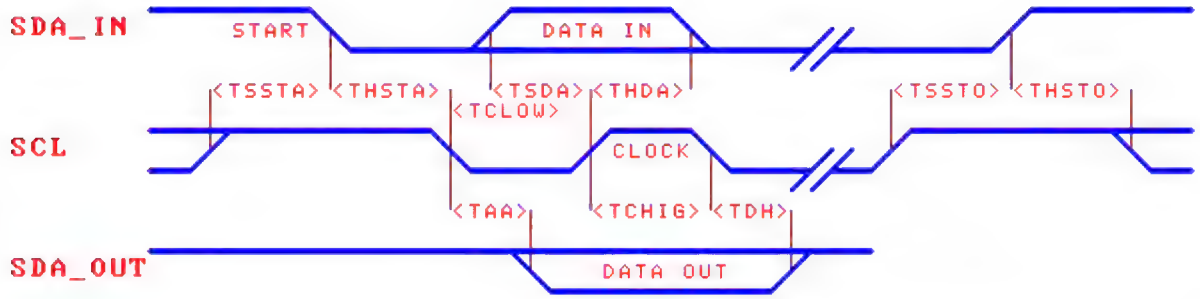
I2C clock register (I2C_CKCFGR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 15 | F/S | RW | Master mode selection:<br>1: Fast mode;<br>0: Standard mode. | 0 |
| 14 | DUTY | RW | Clock HIGH duty cycle in fast mode:<br>1: 36%;<br>0: 33.3%. | 0 |
| [13:12] | Reserved | RO | Reserved. | 0 |
| [11:0] | CCR | RW | Clock frequency division factor register, which decides frequency wave of SCL clock.<br>In I2C standard mode or SMBus mode:<br>$T_{high} = CCR * T_{HCLK}$<br>$T_{low} = CCR * T_{HCLK}$<br><br>In I2C fast mode:<br>If DUTY = 0:<br>$T_{high} = CCR * T_{HCLK}$<br>$T_{low} = 2 * CCR * T_{HCLK}$<br>If DUTY = 1: (speed up to 400 kHz)<br>$T_{high} = 9 * CCR * T_{HCLK}$<br>$T_{low} = 16 * CCR * T_{HCLK}$<br><br>Example: In standard mode, to generate a 100kHz SCL frequency: CCR = FREQR/(2*100KHz), with FREQR = 48MHz, THCLK = 1/48MHz, then CCR = 48MHz/(2*100KHz) = 240, and CCR is written to 0xF0. | 0 |

I2C rise time register (I2C_RTR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:6] | Reserved | RO | Reserved. | 0 |
| [5:0] | TRISE | RW | Maximum rise time. These bits set the rise time of SCL in master mode.<br>The maximum rise time=TRISE–single clock cycle.<br>This bit can be set only when PE is cleared. | 000010b |

## 12.11 I2C Timing




| Symbol | Parameter description (TA=25℃, VIO33=3.3V) | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| TSSTA | Setup time of SCL HIGH before SDA falling edge | 0.4 | | | uS |
| THSTA | Hold time of SCL HIGH after SDA falling edge | 0.4 | | | uS |
| TSDA | Setup time of SDA data before SCL rising edge | 0.05 | | | uS |
| THDA | Hold time of SDA data after SCL rising edge | >TCHIG | | | uS |
| TSSTO | Setup time of SCL HIGH before SDA rising edge | 0.4 | | | uS |
| THSTO | Hold time of SCL HIGH after SDA rising edge | 0.8 | | | uS |
| TCLOW | Time of SCL clock LOW | 0.6 | | | uS |
| TCHIG | Time of SCL clock HIGH | 0.5 | | | uS |
| TAA | SCL falling edge to SDA output valid | 0.006 | | 0.4 | uS |
| TDH | SDA output hold time after SCL falling edge | 0.006 | | | uS |
| TR | SCL/SDA input rising edge time | | | 0.2 | uS |

# Chapter 13 Reserved

# Chapter 14 Reserved

# Chapter 15 Analog-to-digital Converter (ADC)

## 15.1 Introduction to ADC

The chip provides a 12-bit successive approximation analog to digital converter (ADC) and up to 16 channels. CH583 and CH582 each supports 14 external signal sources and 2 internal signal sources.

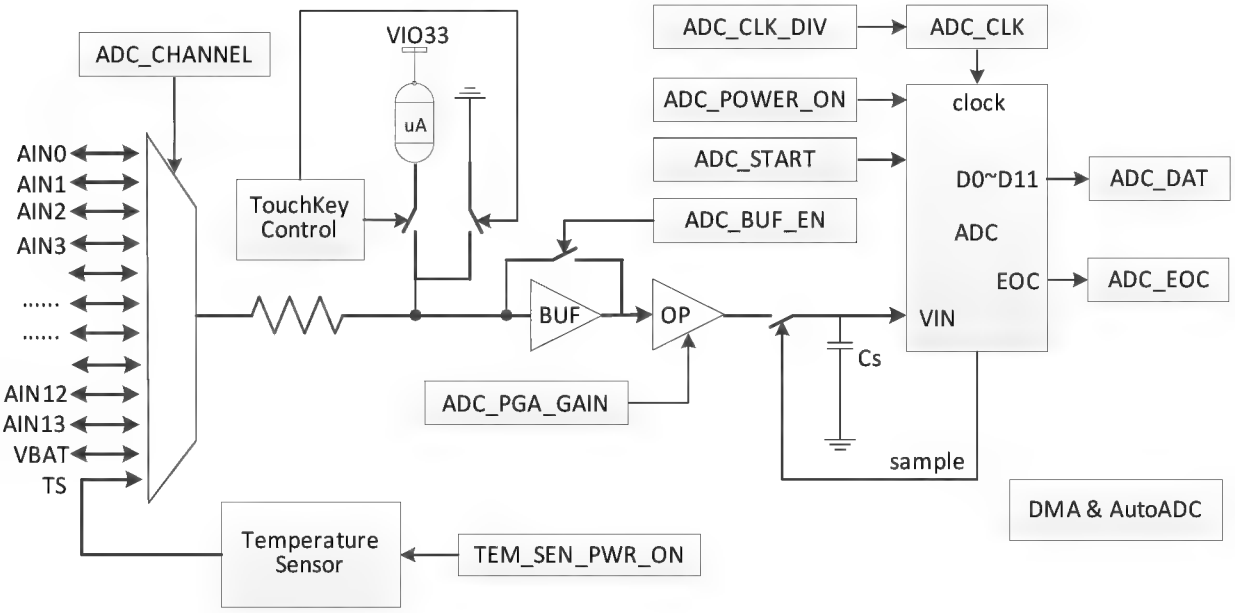
### 15.1.1 Main features

- 12-bit resolution.
- 14external voltage sampling channels, internal temperature detection channels and internal battery voltage detection channels.
- Single-ended input mode and differential input mode detection.
- Optional sampling clock frequency.
- ADC input voltage range: 0V～VIO33.
- Optional PGA: provide gain adjustment options.
- Optional input buffer BUF: support high resistance signal source.
- Support DMA and automatic continuous ADC sampling in timing interval, with adjustable interval.

### 15.1.2 Functional description

The figure below is the block diagram of an ADC module.

Figure 15-1 Architecture diagram of ADC




## 15.2 Register description

Table 15-1 ADC registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_ADC_CHANNEL | 0x40001058 | ADC input channel select register | 0x0F |

| | | | |
|---|---|---|---|
| R8_ADC_CFG | 0x40001059 | ADC configuration register | 0xA0 |
| R8_ADC_CONVERT | 0x4000105A | ADC conversion control register | 0x00 |
| R8_TEM_SENSOR | 0x4000105B | Temperature sensor control register | 0x00 |
| R16_ADC_DATA | 0x4000105C | ADC data register | 0x0XXX |
| R8_ADC_INT_FLAG | 0x4000105E | ADC interrupt flag register | 0x00 |
| R32_ADC_DMA_CTRL | 0x40001060 | DMA control and status register | 0x00 |
| R8_ADC_CTRL_DMA | 0x40001061 | DMA and interrupt control register | 0x00 |
| R8_ADC_DMA_IF | 0x40001062 | ADC and DMA interrupt flag register | 0x00 |
| R8_ADC_AUTO_CYCLE | 0x40001063 | Continuous ADC timing cycle register | 0xXX |
| R16_ADC_DMA_NOW | 0x40001064 | Current buffer address of DMA register | 0x0000XXXX |
| R16_ADC_DMA_BEG | 0x40001068 | Start buffer address of DMA | 0x0000XXXX |
| R16_ADC_DMA_END | 0x4000106C | End buffer address of DMA | 0x0000XXXX |

ADC input channel select register (R8_ADC_CHANNEL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:4] | Reserved | RO | Reserved | 0000b |
| [3:0] | RB_ADC_CH_INX | RW | ADC channel index number, a total of 16 channels:<br>00h-0Dh: External signal channels AIN0~AIN13;<br>0Eh: Battery voltage VBAT;<br>0Fh: Built-in temperature sensor TS. | 1111b |

ADC configuration register (R8_ADC_CFG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | RB_ADC_CLK_DIV | RW | ADC clock frequency selection:<br>00: CK32M divided by 10, 3.2MHz;<br>01: CK32M divided by 4, 8MHz;<br>10: CK32M divided by 6, 5.33MHz;<br>11: CK32M divided by 8, 4MHz; The actual sampling rate is approximately 1/18 of the clock frequency. | 10b |
| [5:4] | RB_ADC_PGA_GAIN | RW | ADC input PGA gain selection:<br>00: -12dB, 1/4;<br>01: -6dB, 1/2;<br>10: 0dB, 1 time, no gain;<br>11: 6dB, 2 times. | 10b |
| 3 | RB_ADC_OFS_TEST | RW | ADC offset error test mode:<br>1: Test/calibrate mode. The lower 12-bit data of data register R16_ADC_DATA will be inverted bit by bit in test mode (0x0573 is inverted to 0x0A8C);<br>0: Normal mode. | 0 |
| 2 | RB_ADC_DIFF_EN | RW | ADC channel signal input mode:<br>1: Differential input;<br>0: Single-ended input. | 0 |

| | | | | |
|---|---|---|---|---|
| 1 | RB_ADC_BUF_EN | RW | ADC input buffer BUF enable:<br>1: Enable; 0: Disable. | 0 |
| 0 | RB_ADC_POWER_ON | RW | ADC module power enable:<br>1: Enable; 0: Disable. | 0 |

Table 15-2 Table of PGA gain selection and input voltage range

| PGA gain selection | The sampled voltage, Vi, calculated from the data converted by ADC | Upper limit of theoretically measurable voltage | Theoretically measurable voltage range (Assuming Vref=1.05V) | Recommended range of actually available voltage |
|---|---|---|---|---|
| -12dB(1/4) | (ADC/512-3)*Vref | 5*Vref | -0.2V ~ VIO33+0.2V | 2.9V ~ VIO33 |
| -6dB(1/2) | (ADC/1024-1)*Vref | 3*Vref | -0.2V ~ 3.15V | 1.9V ~ 3V |
| 0dB(1) | (ADC/2048)*Vref | 2*Vref | 0V ~ 2.1V | 0V ~ 2V |
| 6dB(2) | (ADC/4096+0.5)*Vref | 1.5*Vref | 0.525V ~ 1.575V | 0.6V ~ 1.5V |

ADC: The digital quantity after ADC conversion, namely R16_ADC_DATA.

Vref: The actual voltage value of the power node VINTA of the internal analog circuit is usually 1.05V±0.015V.

Note: If a low voltage is sampled after a negative gain (signal reduction), a large error may be caused in a voltage range. If a high voltage is sampled after a positive gain (signal amplification), ADC conversion value may overflow, so it is recommended to select a reasonable gain mode according to the voltage range of the measured signal.

It is recommended to enable the input buffer by default. Only when the internal resistance of the external signal source is less than 1KΩ, the input buffer can be turned off for ADC.

When using differential input, it is recommended to turn off the input buffer. When used for TouchKey detection, the input buffer must be enabled, and it is recommended to select one of 2 gains of 0dB (priority) or -6dB.

ADC conversion control register (R8_ADC_CONVERT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_ADC_EOC_X | RO | ADC conversion end flag (asynchronous signal):<br>1: Completed; 0: In progress. | 0 |
| [6:1] | Reserved | RO | Reserved | 000000b |
| 0 | RB_ADC_START | RW | ADC conversion enable control and status. It is cleared automatically when non-continuous ADC ends or when DMA ends:<br>1: Start conversion/being converted;<br>0: Stop conversion. | 0 |

Temperature sensor control register (R8_TEM_SENSOR)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_TEM_SEN_PWR_ON | RW | TS temperature sensor power ON:<br>1: ON; 0: OFF. | 0 |
| [6:0] | Reserved | RW | Reserved | 0000000b |

ADC data register (R16_ADC_DATA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:12] | Reserved | RW | Reserved. | 0000b |
| [11:0] | RB_ADC_DATA | RO | Data after ADC conversion. | XXXh |

ADC interrupt flag register (R8_ADC_INT_FLAG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_ADC_IF_EOC | RO | ADC conversion finished flag.<br>This flag can be cleared by writing to register R8_ADC_CONVERT or R8_TKEY_CONVERT. | 0 |
| [6:0] | Reserved | RW | Reserved | 0000000b |

DMA and interrupt control register (R8_ADC_CTRL_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_ADC_AUTO_EN | RW | Automatic continuous ADC sampling enable at timing interval:<br>1: Enable automatic ADC;<br>0: Disable automatic ADC. | 0 |
| 6 | RB_ADC_CONT_EN | RW | ADC continuous conversion mode enable:<br>1: Enable continuous ADC;<br>0: Disable continuous ADC. | 0 |
| 5 | Reserved | RO | Reserved | 00b |
| 4 | RB_ADC_IE_EOC | RW | ADC conversion completion interrupt enable:<br>1: Enable interrupt;<br>0: Disable interrupt. | 0 |
| 3 | RB_ADC_IE_DMA_END | RW | DMA end interrupt enable:<br>1: Enable interrupt;<br>0: Disable interrupt. | 0 |
| 2 | RB_ADC_DMA_LOOP | RW | DMA address loop enable:<br>1: Enable address loop; 0: Disable address loop.<br>If the DMA address loop is enabled, when the DMA address is added to the set end address, it will automatically loop to the start address that is set. | 0 |
| 1 | Reserved | RO | Reserved | 0 |
| 0 | RB_ADC_DMA_ENABLE | RW | DMA function enable:<br>1: Enable DMA;<br>0: Disable DMA. | 0 |

ADC and DMA interrupt flag register (R8_ADC_DMA_IF)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:5] | Reserved | RO | Reserved | 000b |
| 4 | RB_ADC_IF_END_ADC | RW1 | ADC conversion completion interrupt flag, write 1 to clear or when data is taken by DMA or write to register R8_ADC_CONVERT to | 0 |

| | | | | |
|---|---|---|---|---|
| | | | clear this flag:<br>1: ADC has been completed once;<br>0: Not completed. | |
| 3 | RB_ADC_IF_DMA_END | RW1 | DMA completion flag. Write 1 to reset:<br>1: DMA has been completed;<br>0: Not completed. | 0 |
| [2:0] | Reserved | RO | Reserved | 000b |

Continuous ADC timing cycle register (R8_ADC_AUTO_CYCLE)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_ADC_AUTO_CYCLE | RW | The starting value of continuous ADC timing cycle is counted in unit of 16 system clocks and reloaded after reaching 256.<br>Calculation: timing =(256-R8_ADC_AUTO_CYCLE)*16*Tsys | XXh |

DMA current buffer address (R16_ADC_DMA_NOW)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_ADC_DMA_NOW | RO | Current address of DMA data buffer.<br>It can be used to calculate the number of conversions. Calculation: COUNT=(ADC_DMA_NOW-ADC_DMA_BEG)/2. | XXXXh |

DMA start buffer address (R16_ADC_DMA_BEG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_ADC_DMA_BEG | RW | The start address of DMA data buffer, only the lower 15 bits are valid, and the address must be 2-byte aligned. | XXXXh |

DMA end buffer address (R16_ADC_DMA_END)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_ADC_DMA_END | RW | The end address of DMA data buffer (not included), only the lower 15 bits are valid, and the address must be 2-byte aligned. | XXXXh |

## 15.3 ADC configuration

### 15.3.1 External channel sampling

(1) Set RB_ADC_POWER_ON in R8_ADC_CFG to 1 to enable ADC, select the sampling frequency through RB_ADC_CLK_DIV, and enable the input buffer and select signal gain through RB_ADC_BUF_EN and RB_ADC_PGA_GAIN;

(2) Set R8_ADC_CHANNEL register, and select external or internal signal channel;

(3) Set the R8_ADC_CONVERT register, set RB_ADC_START, and enable ADC conversion;

(4) Query and wait for RB_ADC_START to be automatically cleared or wait for RB_ADC_IF_EOC to be set to 1, indicating that the conversion is completed. 12-bit ADC conversion data can be obtained by reading R16_ADC_DATA. If time is enough, it is recommended to convert again and discard the ADC data obtained first;

(5) Repeat steps 2, 3 and 4 to continue to sample another channel or the next set of data;

(6) Single ADC conversion cycle: ADC sampling (4 clocks) + switching 1 clock + conversion time (12 clocks) ≈ 17 Tadc, and 1 time interval is added for continuous ADC. Among them, Tadc = $T_{CK32M}$/RB_ADC_CLK_DIV;

(7) If differential input is used:

Enable differential, select 0# channel: it actually performs differential conversion on the voltage of AIN0 (positive end) and AIN2 (negative end);

Enable differential, select 1# channel: it actually performs differential conversion on the voltage of AIN1 (positive end) and AIN3 (negative end);

As a result of ADC conversion, if the data is greater than 0x800, it means that the voltage of the differential positive end is higher than the voltage of the differential negative end. If the data is less than 0x800, it means the voltage of the differential positive end is lower than the voltage of the differential negative end. Taking the PGA gain selection of 0dB as an example, the theoretically measurable voltage ranges from -1.05V to 1.05V, and 0x400 means that the voltage of the differential positive end is lower than the voltage of the differential negative end by about 0.5 Vref.

### 15.3.2 Temperature sensor sampling

(1) Set the RB_TEM_SEN_PWR_ON in R8_TEM_SENSOR register to 1 to enable the temperature sensor, set R8_ADC_CHANNEL to 15, and select the temperature sensor signal and connect to ADC input;

(2) Set RB_ADC_POWER_ON to 1 to enable ADC, set RB_ADC_DIFF_EN to 1, set RB_ADC_CLK_DIV, set RB_ADC_BUF_EN to 0, and set RB_ADC_PGA_GAIN to 11;

(3) Set R8_ADC_CONVERT register, set RB_ADC_START to 1, and enable ADC conversion;

(4) Query and wait for RB_ADC_START to be automatically cleared or wait for RB_ADC_IF_EOC to be set to 1, read R16_ADC_DATA to obtain 12-bit ADC conversion data. In case of high requirements for accuracy, it is recommended to repeat steps 3 and 4 for several times to calculate the average value of ADC data;

(5) The temperature value is obtained according to the conversion relationship between voltage and temperature. Please refer to the evaluation board example program for details.

### 15.3.3 Enable DMA automatic continuous ADC

(1) Set ADC parameters and select channels with reference to non-DMA mode;

(2) Set R8_ADC_AUTO_CYCLE to select continuous ADC cycle;

(3) Set the R16_ADC_DMA_BEG register as the start address of buffer that stores ADC data, set the R16_ADC_DMA_END register as the end address of buffer that stores ADC data (not included), and set the RB_ADC_DMA_ENABLE in R8_ADC_CTRL_DMA to 1, to enable DMA function;

(4) Optional step. If it is needed to enable interrupts, set the corresponding interrupt enable register bit. When RB_ADC_IE_EOC=1, the ADC will be triggered by RB_ADC_IF_END_ADC to complete the interrupt. When RB_ADC_IE_EOC=0 and RB_ADC_IE_DMA_END=0, the ADC will be triggered by RB_ADC_IF_EOC to complete the interrupt;

(5) Set RB_ADC_AUTO_EN to 1 to enable automatic continuous ADC;

(6) RB_ADC_IF_EOC and RB_ADC_IF_END_ADC is set to 1 after ADC is completed, but RB_ADC_IF_END_ADC is automatically cleared after the data is taken by DMA, so you can query RB_ADC_IF_EOC to query ADC completion status.

# Chapter 16 Touch-Key

## 16.1 Introduction to Touch-Key

CH583 and CH582 each provides a capacitance detection module, which can be used with the ADC module to realize the detection function of capacitance Touch-Key. There are a total of 14 input channels, and the supported capacitance value of Touch-Key ranges from 10pF to 100pF. The driver shielding output is provided to improve sensitivity.

## 16.2 Register description

Table 16-1 Touch-Key registers

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_TKEY_COUNT | 0x40001054 | TouchKey charge/discharge time register | 0xXX |
| R8_TKEY_CONVERT | 0x40001056 | TouchKey detection control register | 0x00 |
| R8_TKEY_CFG | 0x40001057 | TouchKey configuration register | 0x00 |

TouchKey charge/discharge time register (R8_TKEY_COUNT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:5] | RB_TKEY_DISCH_CNT | RW | The number of Touch-Key discharge cycles, counting in unit of ADC clock selected by RB_ADC_CLK_DIV.<br>Calculation: discharge time = (RB_TKEY_DISCH_CNT+1)*Tadc. | XXXb |
| [4:0] | RB_TKEY_CHARG_CNT | RW | The number of Touch-Key charge cycle, counting in unit of ADC clock selected by RB_ADC_CLK_DIV. Calculation: charge time = (RB_TKEY_CHARG_CNT+4)*Tadc. | XXXXXb |

TouchKey detection control register (R8_TKEY_CONVERT)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:1] | Reserved | RO | Reserved | 0000000b |
| 0 | RB_TKEY_START | RW | TouchKey detection control and status, automatically cleared:<br>1: Start to detect/being converted;<br>0: Stop converting. | 0 |

TouchKey configuration register (R8_TKEY_CFG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:4] | Reserved | RO | Reserved | 0000b |
| 3 | RB_TKEY_PGA_ADJ | RW | Select the operating speed of PGA in ADC:<br>1: High speed but slightly larger power | 0 |

| | | | | |
|---|---|---|---|---|
| | | | consumption;<br>0: Normal speed. | |
| 2 | RB_TKEY_DRV_EN | RW | TouchKey driver shielding enable:<br>1: Enable; 0: Disable. | 0 |
| 1 | RB_TKEY_CURRENT | RO | TouchKey charge current selection:<br>1: 60% current;<br>0: Rated current. | 0 |
| 0 | RB_TKEY_PWR_ON | RW | TouchKey module power ON:<br>1: ON;<br>0: OFF. | 0 |

## 16.3 Touch-Key configuration

Please refer to and call related subprograms.

# Chapter 17 USB Controller

## 17.1 Introduction to USB controller

CH583 and CH582 each is embedded with 2 separate USB host-slave controllers and transceivers. The features are:

- USB Host function and USB Device function.
- USB2.0 full-speed 12Mbps and low-speed 1.5Mbps.
- USB control transmission, bulk transmission, interrupt transmission, synchronous/real-time transmission.
- Data packets up to 64 bytes, built-in FIFO, support interrupt and DMA.

## 17.2 Register description

The base addresses of the 2 USB controllers are 0x40008000 and 0x40008400. The others are the same, take USB as example in this datasheet, please refer this datasheet for USB2.

USB related registers are divided to 3 categories, some of which are multiplexed in host and device mode.

(1) USB global registers.

(2) USB device control registers.

(3) USB host control registers.

### 17.2.1 Description of global registers

Table 17-1 USB registers (those marked in grey are controlled by RB_UC_RESET_SIE reset)

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_USB_CTRL | 0x40008000 | USB control register | 0x06 |
| R8_USB_INT_EN | 0x40008002 | USB interrupt enable register | 0x00 |
| R8_USB_DEV_AD | 0x40008003 | USB device address register | 0x00 |
| R32_USB_STATUS | 0x40008004 | USB status register | 0xXX20XXXX |
| R8_USB_MIS_ST | 0x40008005 | USB miscellaneous status register | 0xXX |
| R8_USB_INT_FG | 0x40008006 | USB interrupt flag register | 0x20 |
| R8_USB_INT_ST | 0x40008007 | USB interrupt status register | 0xXX |
| R8_USB_RX_LEN | 0x40008008 | USB receiving length register | 0xXX |

USB control register (R8_USB_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UC_HOST_MODE | RW | USB working mode selection:<br>1: HOST; 0: DEVICE. | 0 |
| 6 | RB_UC_LOW_SPEED | RW | USB bus signal transmission rate selection:<br>1: 1.5Mbps; 0: 12Mbps. | 0 |
| 5 | RB_UC_DEV_PU_EN | RW | USB device enable and internal pull-up resistor control in USB device mode. If it is 1, USB device transmission is enabled and the internal pull-up resistor is also enabled. RB_PIN_USB_DP_PU can replace this bit. | 0 |
| [5:4] | MASK_UC_SYS_CTRL | RW | See the table below to configure the USB | 00b |

| | | | | |
|---|---|---|---|---|
| | | | system. | |
| 3 | RB_UC_INT_BUSY | RW | Auto pause enable before USB transmission completion interrupt flag is not cleared:<br>1: It will automatically pause before the interrupt flag UIF_TRANSFER is not cleared. In device mode, it will automatically respond to busy NAK. In host mode, it will automatically pause subsequent transmission;<br>0: Not pause. | 0 |
| 2 | RB_UC_RESET_SIE | RW | Software reset control of USB protocol processor:<br>1: Force to reset the USB protocol processor (SIE), software is needed to cleared;<br>0: Not reset. | 1 |
| 1 | RB_UC_CLR_ALL | RW | USB FIFO and interrupt flag clear:<br>1: Force to empty and clear;<br>0: Not clear. | 1 |
| 0 | RB_UC_DMA_EN | RW | USB DMA and DMA interrupt control:<br>1: Enable the DMA function and DMA interrupt;<br>0: Disable DMA. | 0 |

RB_UC_HOST_MODE and MASK_UC_SYS_CTRL constitute the USB system control combination:

| RB_UC_HOST_MODE | MASK_UC_SYS_CTRL | USB system control description |
|---|---|---|
| 0 | 00 | Disable USB device function and disable the internal pull-up resistor.<br>*Note: If RB_PIN_USB_DP_PU=1, the DP pull-up resistor will be enabled by force.* |
| 0 | 01 | Enable USB device function and disable the internal pull-up resistor, the external pull-up is needed.<br>*Note: If RB_PIN_USB_DP_PU=1, the DP pull-up resistor will be enabled by force.* |
| 0 | 1x | Enable USB device function and enable the internal 1.5K pull-up resistor. The pull-up resistor has priority over the pull-down resistor, and can also be used in GPIO mode. |
| 1 | 00 | USB host mode, in normal working status. |
| 1 | 01 | USB host mode, DP/DM is forced to output SE0 status. |
| 1 | 10 | USB host mode, DP/DM is forced to output J status. |
| 1 | 11 | USB host mode, DP/DM is forced to output K status/wake-up. |

USB interrupt enable register (R8_USB_INT_EN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UIE_DEV_SOF | RW | In USB device mode, receive SOF packet interrupt: | 0 |

| | | | | |
|---|---|---|---|---|
| | | | 1: Enable interrupt; 0: Disable interrupt. | |
| 6 | RB_UIE_DEV_NAK | RW | In USB device mode, receive NAK interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_UIE_FIFO_OV | RW | FIFO overflow interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 3 | RB_UIE_HST_SOF | RW | In the USB host mode, SOF timing interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 2 | RB_UIE_SUSPEND | RW | USB bus suspend or wake-up event interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 1 | RB_UIE_TRANSFER | RW | USB transfer completion interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| 0 | RB_UIE_DETECT | RW | In USB host mode, USB device connection or disconnection event interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |
| | RB_UIE_BUS_RST | RW | In USB device mode; USB bus reset event interrupt:<br>1: Enable interrupt; 0: Disable interrupt. | 0 |

USB device address register (R8_USB_DEV_AD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UDA_GP_BIT | RW | USB general flag, user-defined. | 0 |
| [6:0] | MASK_USB_ADDR | RW | Host mode: address of USB device currently operated;<br>Device mode: the address of the USB itself. | 0000000b |

USB miscellaneous status register (R8_USB_MIS_ST)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UMS_SOF_PRES | RO | SOF packet indication status in USB host mode:<br>1: SOF packet will be sent, and it will be automatically delayed if there are other USB data packets;<br>0: No SOF package is sent. | X |
| 6 | RB_UMS_SOF_ACT | RO | SOF packet transmission status in USB host mode:<br>1: SOF packet is being sent out;<br>0: The transmission is completed or idle. | X |
| 5 | RB_UMS_SIE_FREE | RO | USB protocol processor free status:<br>1: Free;<br>0: Busy, USB transmission is in progress. | 1 |
| 4 | RB_UMS_R_FIFO_RDY | RO | USB receiver FIFO data ready status:<br>1: Receiver FIFO not empty;<br>0: Receiver FIFO empty. | 0 |
| 3 | RB_UMS_BUS_RESET | RO | USB bus reset status: | X |

| | | | | |
|---|---|---|---|---|
| | | | 1: The current USB bus is at the reset status;<br>0: The current USB bus is at the non-reset status. | |
| 2 | RB_UMS_SUSPEND | RO | USB suspend status:<br>1: USB bus is in suspended status, and there is no USB activity for a period of time;<br>0: USB bus is in non-suspended status. | 0 |
| 1 | RB_UMS_DM_LEVEL | RO | In USB host mode, the level status of the DM pin when the device is just connected to the USB port, used to determine speed:<br>1: High level/ low-speed;<br>0: Low level/ full-speed. | 0 |
| 0 | RB_UMS_DEV_ATTACH | RO | USB device connection status of the port in USB host mode:<br>1: The port has been connected to a USB device;<br>0: No USB device is connected to the port. | 0 |

USB interrupt flag register (R8_USB_INT_FG)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_U_IS_NAK | RO | In USB device mode, NAK acknowledge status:<br>1: NAK acknowledge during current USB transmission;<br>0: No NAK acknowledge. | 0 |
| 6 | RB_U_TOG_OK | RO | Current USB transmission DATA0/1 synchronous flag match status:<br>1: Synchronous; 0: Asynchronous. | 0 |
| 5 | RB_U_SIE_FREE | RO | USB protocol processor idle status:<br>1: USB idle;<br>0: Busy, USB transmission is in progress. | 1 |
| 4 | RB_UIF_FIFO_OV | RW | USB FIFO overflow interrupt flag. Write 1 to reset:<br>1: FIFO overflow trigger; 0: No event. | 0 |
| 3 | RB_UIF_HST_SOF | RW | SOF timing interrupt flag in USB host mode. Write 1 to reset:<br>1: SOF packet transmission completion trigger;<br>0: No event. | 0 |
| 2 | RB_UIF_SUSPEND | RW | USB bus suspend or wake-up event interrupt flag. Write 1 to reset:<br>1: USB suspend event or wake-up event trigger;<br>0: No event. | 0 |
| 1 | RB_UIF_TRANSFER | RW | USB transmission completion interrupt flag. Write 1 to reset:<br>1: USB transmission completion trigger;<br>0: No event. | 0 |
| 0 | RB_UIF_DETECT | RW | In USB host mode, the USB device connection or | 0 |

| | | | | |
|---|---|---|---|---|
| | | | disconnection event interrupt flag. Write 1 to reset:<br>1: USB device connection or disconnection trigger is detected;<br>0: No event. | |
| | RB_UIF_BUS_RST | RW | USB bus reset event interrupt flag in USB device mode. Write 1 to reset:<br>1: USB bus reset event trigger;<br>0: No event. | 0 |

USB interrupt status register (R8_USB_INT_ST)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UIS_SETUP_ACT | RO | In USB device mode, when this bit is 1, 8-byte SETUP request packet has been successfully received. SETUP token does not affect RB_UIS_TOG_OK, MASK_UIS_TOKEN, MASK_UIS_ENDP and R8_USB_RX_LEN. | 0 |
| 6 | RB_UIS_TOG_OK | RO | Current USB transmission DATA0/1 synchronization flag matching status. The same as RB_U_TOG_OK:<br>1: Synchronous;<br>0: Asynchronous. | 0 |
| [5:4] | MASK_UIS_TOKEN | RO | In device mode, the token PID of the current USB transfer transaction. | XXb |
| [3:0] | MASK_UIS_ENDP | RO | In device mode, the endpoint number of the current USB transfer transaction. | XXXXb |
| | MASK_UIS_H_RES | RO | In host mode, the response PID identification of the current USB transfer transaction.<br>0000: The device has no response or timeout;<br>Other values: Respond PID. | XXXXb |

MASK_UIS_TOKEN is used to identify the token PID of the current USB transfer transaction in USB device mode: 00 represents OUT packet; 01 represents SOF packet; 10 represents IN packet; 11 represents idle.

When MASK_UIS_TOKEN is not idle, and RB_UIS_SETUP_ACT is also 1, it is required to process the former first, and clear RB_UIF_TRANSFER once after the former is processed to make the former enter the idle status, and then process the latter, and finally clear RB_UIF_TRANSFER again.

MASK_UIS_H_RES is only valid in host mode. In host mode, if the host sends OUT/SETUP token packet, the PID will be the handshake packet ACK/NAK/STALL, or that device has no response/timeout. If the host sends IN token packet, the PID will the PID of the data packet (DATA0/DATA1) or the handshake packet PID.

USB receiving length register (R8_USB_RX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_USB_RX_LEN | RO | The number of data bytes received by the current USB endpoint. | XXh |

### 17.2.2 Description of device registers

In USB device mode, the chip provides 8 sets of bidirectional endpoints, including endpoint0, endpoint1, endpoint2, endpoint3, endpoint4, endpoint5, endpoint6 and endpoint7. The maximum data packet length of each endpoint is 64 bytes.

Endpoint0 is the default endpoint and supports control transmission. The transmission and reception share a 64-byte data buffer.

Endpoint1, endpoint2 and endpoint3 each includes a transmission endpoint IN and a reception endpoint OUT.

The reception and the transmission each has a separate 64-byte or double 64-byte data buffer, supporting bulk transmission, interrupt transmission, and real-time/synchronous transmission.

Endpoint4, endpoint5, endpoint6 and endpoint7 each includes a transmission endpoint IN and a reception endpoint OUT. The reception and the transmission each has a separate 64-byte data buffer, supporting bulk transmission, interrupt transmission, and real-time/synchronous transmission.

Each set of endpoint has a control register R8_UEPn_CTRL and a transmit length register R8_UEPn_T_LEN (n=0/1/2/3/4/5/6/7), used to set the synchronization trigger bit of endpoint, the response to OUT transactions and IN transactions and the length of data to be sent.

As the necessary USB bus pull-up resistor of USB device, it can be set whether to be enabled by software at any time. When RB_UC_DEV_PU_EN of USB control register R8_USB_CTRL is set to 1, the controller will set according to the speed of RB_UD_LOW_SPEED, internally connect the pull-up resistor with the DP/DM pin of the USB bus and enable the USB device function. The above control cannot be used in sleep or power-down mode, but RB_PIN_USB_DP_PU of R16_PIN_ANALOG_IE can enable the pull-up resistor of DP pin by force without being affected by sleep mode.

When a USB bus reset or USB bus suspend or wake-up event is detected, or when the USB successfully processes data sending or receiving, the USB protocol processor will set corresponding interrupt flag. If the interrupt enabling is switched on, the corresponding interrupt request will also be generated. The application program can directly query or query and analyze the interrupt flag register R8_USB_INT_FG in the USB interrupt service program, and perform corresponding processing according to RB_UIF_BUS_RST and RB_UIF_SUSPEND. In addition, if RB_UIF_TRANSFER is valid, it is required to continue to analyze the USB interrupt state register R8_USB_INT_ST, and perform the corresponding processing according to the current endpoint number MASK_UIS_ENDP and the current transaction token PID identification MASK_UIS_TOKEN. If the synchronization trigger bit RB_UEP_R_TOG of OUT transaction of each endpoint is set in advance, RB_U_TOG_OK or RB_UIS_TOG_OK can be used to judge whether the synchronization trigger bit of the data packet received matches the synchronization trigger bit of the endpoint. If the data is synchronous, the data is valid. If the data is not synchronous, the data should be discarded. Every time the USB sending or receiving interrupt is processed, the synchronization trigger bit of corresponding endpoint should be modified correctly to detect whether the data packet sent next time and the data packet received next time are synchronous. In addition, RB_UEP_AUTO_TOG can be set to automatically flip the corresponding synchronization trigger bit after sending or receiving successfully.

The data to be sent by each endpoint is in their own buffer, and the length of the data to be sent is independently set in R8_UEPn_T_LEN. The data received by each endpoint is in their own buffer, but the length of the data received

is in the USB length receiving register R8_USB_RX_LEN, and it can be distinguished according to the current endpoint number when USB receives an interrupt.

Table 17-2 USB device registers (those marked in grey are controlled by RB_UC_RESET_SIE reset)

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UDEV_CTRL | 0x40008001 | USB device physical port control register | 0xX0 |
| R8_UEP4_1_MOD | 0x4000800c | Endpoint 1/4 mode control register | 0x00 |
| R8_UEP2_3_MOD | 0x4000800d | Endpoint 2/3 mode control register | 0x00 |
| R8_UEP567_MOD | 0x4000800e | Endpoint 5/6/7 mode control register | 0x00 |
| R16_UEP0_DMA | 0x40008010 | Start address of endpoint0 buffer | 0xXXXX |
| R16_UEP1_DMA | 0x40008014 | Start address of endpoint1 buffer | 0xXXXX |
| R16_UEP2_DMA | 0x40008018 | Start address of endpoint2 buffer | 0xXXXX |
| R16_UEP3_DMA | 0x4000801c | Start address of endpoint3 buffer | 0xXXXX |
| R8_UEP0_T_LEN | 0x40008020 | Endpoint0 transmission length register | 0xXX |
| R8_UEP0_CTRL | 0x40008022 | Endpoint0 control register | 0x00 |
| R8_UEP1_T_LEN | 0x40008024 | Endpoint1 transmission length register | 0xXX |
| R8_UEP1_CTRL | 0x40008026 | Endpoint1 control register | 0x00 |
| R8_UEP2_T_LEN | 0x40008028 | Endpoint2 transmission length register | 0xXX |
| R8_UEP2_CTRL | 0x4000802a | Endpoint2 control register | 0x00 |
| R8_UEP3_T_LEN | 0x4000802c | Endpoint3 transmission length register | 0xXX |
| R8_UEP3_CTRL | 0x4000802e | Endpoint3 control register | 0x00 |
| R8_UEP4_T_LEN | 0x40008030 | Endpoint4 transmission length register | 0xXX |
| R8_UEP4_CTRL | 0x40008032 | Endpoint4 control register | 0x00 |
| R16_UEP5_DMA | 0x40008054 | Start address of endpoint5 buffer | 0xXXXX |
| R16_UEP6_DMA | 0x40008058 | Start address of endpoint6 buffer | 0xXXXX |
| R16_UEP7_DMA | 0x4000805c | Start address of endpoint7 buffer | 0xXXXX |
| R8_UEP5_T_LEN | 0x40008064 | Endpoint5 transmission length register | 0xXX |
| R8_UEP5_CTRL | 0x40008066 | Endpoint5 control register | 0x00 |
| R8_UEP6_T_LEN | 0x40008068 | Endpoint6 transmission length register | 0xXX |
| R8_UEP6_CTRL | 0x4000806a | Endpoint6 control register | 0x00 |
| R8_UEP7_T_LEN | 0x4000806c | Endpoint7 transmission length register | 0xXX |
| R8_UEP7_CTRL | 0x4000806e | Endpoint7 control register | 0x00 |

USB device physical port control register (R8_UDEV_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UD_PD_DIS | RW | Internal pull-down resistor control of UD+/UD- pin of USB device port:<br>1: Disable internal pull-down; 0: Enable internal pull-down.<br>It also can be used in GPIO mode to provide pull-down resistor. | 1 |

| | | | | |
|---|---|---|---|---|
| 6 | Reserved | RO | Reserved | 0 |
| 5 | RB_UD_DP_PIN | RO | Current UD + pin status:<br>1: High level; 0: Low level. | X |
| 4 | RB_UD_DM_PIN | RO | Current UD- pin status:<br>1: High level;<br>0: Low level. | X |
| 3 | Reserved | RO | Reserved | 0 |
| 2 | RB_UD_LOW_SPEED | RW | Current UD- pin status: 1: High level;<br>0: Low level. | 0 |
| 1 | RB_UD_GP_BIT | RW | USB device mode general flag, user-defined. | 0 |
| 0 | RB_UD_PORT_EN | RW | USB device physical port enable:<br>1: Enable;<br>0: Disable. | 0 |

Endpoint 1/4 mode control register (R8_UEP4_1_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UEP1_RX_EN | RW | 1: Enable endpoint 1 reception (OUT);<br>0: Disable endpoint 1 reception. | 0 |
| 6 | RB_UEP1_TX_EN | RW | 1: Enable endpoint 1 transmission (IN);<br>0: Disable endpoint 1 transmission. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_UEP1_BUF_MOD | RW | Endpoint 1 data buffer mode control bit.。 | 0 |
| 3 | RB_UEP4_RX_EN | RW | 1: Enable endpoint 4 reception (OUT);<br>0: Disable endpoint 4 reception. | 0 |
| 2 | RB_UEP4_TX_EN | RW | 1: Enable endpoint 4 transmission (IN);<br>0: Disable endpoint 4 transmission. | 0 |
| [1:0] | Reserved | RO | Reserved | 00b |

The data buffer modes of USB endpoint0 and endpoint4 are configured by a combination of bUEP4_RX_EN and bUEP4_TX_EN. Refer to the following table for details:

Table 17-3 Endpoint0/4 buffer modes

| bUEP4_RX_EN | bUEP4_TX_EN | Description: arrange from low to high with UEP0 DMA as the start address |
|---|---|---|
| 0 | 0 | Endpoint0 single 64-byte transmit/receive shared buffers (IN and OUT). |
| 1 | 0 | Endpoint0 single 64-byte transmit/receive shared buffers; endpoint4 single 64-byte reception buffers (OUT). |
| 0 | 1 | Endpoint0 single 64-byte transmit/receive shared buffers; endpoint4 single 64-byte transmission buffers (IN). |
| 1 | 1 | Endpoint0 single 64-byte transmit/receive shared buffers; endpoint4 single 64-byte reception buffers (OUT);<br>Endpoint4 single 64-byte receive buffer (IN). All 192 bytes are arranged as follows:<br>UEP0_DMA+0 address: 64-byte start address of endpoint0 transmit/receive shared buffer; |

| | | |
|---|---|---|
| | | UEP0_DMA+64 address: 64-byte start address of endpoint4 receive buffer;<br>UEP0_DMA+128 address: 64-byte start address of endpoint4 transmit buffer. |

Endpoint 2/3 mode control register (R8_UEP2_3_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UEP3_RX_EN | RW | 1: Enable endpoint 3 reception (OUT);<br>0: Disable endpoint 3 reception. | 0 |
| 6 | RB_UEP3_TX_EN | RW | 1: Enable endpoint 3 transmission (IN);<br>0: Disable endpoint 3 transmission. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_UEP3_BUF_MOD | RW | Endpoint 3 data buffer mode control bit. | 0 |
| 3 | RB_UEP2_RX_EN | RW | 1: Enable endpoint 2 reception (OUT);<br>0: Disable endpoint 2 reception. | 0 |
| 2 | RB_UEP2_TX_EN | RW | 1: Enable endpoint 2 transmission (IN);<br>0: Disable endpoint 2 transmission. | 0 |
| 1 | Reserved | RO | Reserved | 0 |
| 0 | RB_UEP2_BUF_MOD | RW | Endpoint 2 data buffer mode control bit. | 0 |

The data buffer modes of USB endpoint1/2/3 are controlled by a combination of RB_UEPn_RX_EN, RB_UEPn_TX_EN and RB_UEPn_BUF_MOD(n=1/2/3) respectively, refer to the following table for details. Among them, in the double 64-byte buffer mode, the first 64-byte buffer will be selected based on RB_UEP_*_TOG=0 and the last 64-byte buffer will be selected based on RB_UEP_*_TOG=1 during USB data transmission, and RB_UEP_AUTO_TOG=1 is set to realize automatic switch.

Table 17-4 Endpoint n buffer modes (n=1/2/3)

| RB_UEPn_RX_EN | RB_UEPn_TX_EN | RB_UEPn_BUF_MOD | Description: Arrange from low to high with R16_UEPn_DMA as the start address |
|---|---|---|---|
| 0 | 0 | X | Endpoint is disabled, and R16_UEPn_DMA buffer is not used. |
| 1 | 0 | 0 | Single 64-byte receive buffer (OUT). |
| 1 | 0 | 1 | Double 64-byte receive buffer (OUT), selected by RB_UEP_R_TOG. |
| 0 | 1 | 0 | Single 64-byte transmit buffer (IN). |
| 0 | 1 | 1 | Double 64-byte transmit buffer (IN), selected by RB_UEP_T_TOG. |
| 1 | 1 | 0 | Single 64-byte receive buffer (OUT), single 64-byte transmit buffer (IN). |
| 1 | 1 | 1 | Double 64-byte receive buffer (OUT), selected by RB_UEP_R_TOG.<br>Double 64-byte transmit buffer (IN), selected by RB_UEP_T_TOG.<br>All 256 bytes are arranged as follows:<br>UEPn_DMA+0 address: endpoint receive address when |

| | | | |
|---|---|---|---|
| | | | RB_UEP_R_TOG=0;<br>UEPn_DMA+64 address: endpoint receive address when RB_UEP_R_TOG=1;<br>UEPn_DMA+128 address: endpoint transmit address when RB_UEP_T_TOG=0;<br>UEPn_DMA+192 address: endpoint transmit address when RB_UEP_T_TOG=1. |

Endpoint 5/6/7 mode control register (R8_UEP567_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:6] | Reserved | RO | Reserved | 00b |
| 5 | RB_UEP7_RX_EN | RW | 1: Enable endpoint 7 reception (OUT);<br>0: Disable endpoint 7 reception. | 0 |
| 4 | RB_UEP7_TX_EN | RW | 1: Enable endpoint 7 transmission (IN);<br>0: Disable endpoint 7 transmission. | 0 |
| 3 | RB_UEP6_RX_EN | RW | 1: Enable endpoint 6 reception (OUT);<br>0: Disable endpoint 6 reception. | 0 |
| 2 | RB_UEP6_TX_EN | RW | 1: Enable endpoint 6 transmission (IN);<br>0: Disable endpoint 6 transmission. | 0 |
| 1 | RB_UEP5_RX_EN | RW | 1: Enable endpoint 5 reception (OUT);<br>0: Disable endpoint 5 reception. | 0 |
| 0 | RB_UEP5_TX_EN | RW | 1: Enable endpoint 5 transmission (IN);<br>0: Disable endpoint 5 transmission. | 0 |

The data buffer modes of USB endpoint5, endpoint6 and endpoint7 are controlled by a combination of RB_UEPn_RX_EN and RB_UEPn_TX_EN (n=5/6/7) respectively, refer to the following table for details.

Table 17-5 Endpoint n buffer modes (n=5/6/7)

| RB_UEPn_RX_EN | RB_UEPn_TX_EN | Description: Arrange from low to high with R16_UEPn_DMA as the start address |
|---|---|---|
| 0 | 0 | Endpoint is disabled, and R16_UEPn_DMA buffer is not used. |
| 1 | 0 | Single 64-byte receive buffer (OUT). |
| 0 | 1 | Single 64-byte transmit buffer (IN). |
| 1 | 1 | Single 64-byte receive buffer (OUT), single 64-byte transmit buffer (IN). |

Endpoint n buffer start address (R16_UEPn_DMA) (n=0/1/2/3/5/6/7)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_UEPn_DMA | RW | Endpoint n buffer start address.<br>Lower 15 bits are valid, and the address must be 4-byte aligned. | XXXXh |

*Note: The length of the buffer that receives data >= min (maximum data packet length possibly received + 2 bytes, 64 bytes).*

Endpoint n transmission length register (R8_UEPn_T_LEN) (n=0/1/2/3/4/5/6/7)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UEPn_T_LEN | RW | Set the number of data bytes that USB endpoint n is ready to send. | XXh |

Endpoint n control register (R8_UEPn_CTRL) (n=0/1/2/3/4/5/6/7)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UEP_R_TOG | RW | Expected synchronization trigger bit of the receiver (process OUT transactions) of USB endpoint n:<br>1: Expect DATA1;<br>0: Expect DATA0. | 0 |
| 6 | RB_UEP_T_TOG | RW | Synchronization trigger bit of the transmitter (process IN transactions) of USB endpoint n<br>1: Send DATA1;<br>0: Send DATA0. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_UEP_AUTO_TOG | RW | Synchronization trigger bit auto flip enable control bit:<br>1: After the data is sent or received successfully, the corresponding synchronization trigger bit is automatically flipped;<br>0: Not flipped automatically, but can be switched manually.<br>It only supports endpoint 1/2/3/5/6/7. | 0 |
| [3:2] | MASK_UEP_R_RES | RW | Control on the response to OUT transactions by the receiver of USB endpoint n:<br>00: Respond ACK;<br>01: Timeout/no response, used for real-time/synchronous transmission of non-endpoint 0;<br>10: Respond to NAK or busy;<br>11: Respond to STALL or error. | 00b |
| [1:0] | MASK_UEP_T_RES | RW | Control on response to IN transactions of the transmitter of endpoint n:<br>00: DATA0/DATA1 data is ready and ACK is expected;<br>01: Respond to DATA0/DATA1 and expect no response, used for real-time/synchronous transmission of non-endpoint 0;<br>10: Respond to NAK or busy; | 00b |

| | | | | |
|---|---|---|---|---|
| | | | 11: Respond to STALL or error. | |

### 17.2.3 Description of host registers

In USB host mode, the chip provides 1 set of bidirectional host endpoints, including a transmission endpoint OUT and a reception endpoint IN. The maximum length of data packet is 64 bytes. It supports control transmission, interrupt transmission, bulk transmission and real-time/synchronous transmission.

Each USB transaction initiated by host endpoint always automatically sets the RB_UIF_TRANSFER interrupt flag after the processing ends. The application program can directly query or query and analyze the interrupt flag register (R8_USB_INT_FG) in the USB interrupt service program, and perform corresponding processing according to each interrupt flag. In addition, if RB_UIF_TRANSFER is valid, it is required to continue to analyze the USB interrupt status register (R8_USB_INT_ST), and perform the corresponding processing according to the response PID identification (MASK_UIS_H_RES) of the current USB transmission transaction.

If the synchronization trigger bit (RB_UH_R_TOG) of IN transaction of host reception endpoint is set in advance, whether the synchronization trigger bit of the data packet received matches the synchronization trigger bit of the endpoint can be judged through RB_U_TOG_OK or RB_UIS_TOG_OK. If the data is synchronous, the data is valid. If the data is not synchronous, the data should be discarded. Each time the USB sending or receiving interrupt is processed, the synchronization trigger bit of corresponding host endpoint should be modified correctly to synchronize the data packet sent next time and detect whether the data packet received next time is synchronous. In addition, RB_UH_T_AUTO_TOG and RB_UH_R_AUTO_TOG can be set to automatically flip the corresponding synchronization trigger bit after sending or receiving successfully.

USB host token setting register (R8_UH_EP_PID) is used to set the endpoint number of the target device being operated and the token PID packet identification of the USB transmission transaction. The data corresponding to the SETUP token and OUT token is provided by the host transmission endpoint. The data to be sent is in the R16_UH_TX_DMA buffer, and the length of the data to be sent is set in R16_UH_TX_LEN. The data corresponding to the IN token is returned by the target device to the host reception endpoint, the data received is stored in the R16_UH_RX_DMA buffer, and the length of data received is stored in R8_USB_RX_LEN.

Table 17-6 USB host registers (Those marked in grey are controlled by RB_UC_RESET_SIE reset)

| Name | Access address | Description | Reset value |
|---|---|---|---|
| R8_UHOST_CTRL | 0x40008001 | USB host physical port control register | 0xX0 |
| R8_UH_EP_MOD | 0x4000800d | USB host endpoint mode control register | 0x00 |
| R16_UH_RX_DMA | 0x40008018 | USB host receive buffer start address | 0xXXXX |
| R16_UH_TX_DMA | 0x4000801c | USB host transmit buffer start address | 0xXXXX |
| R8_UH_SETUP | 0x40008026 | USB host auxiliary setting register | 0x00 |
| R8_UH_EP_PID | 0x40008028 | USB host token setting register | 0x00 |
| R8_UH_RX_CTRL | 0x4000802a | USB host reception endpoint control register | 0x00 |
| R8_UH_TX_LEN | 0x4000802c | USB host transmission length register | 0xXX |
| R8_UH_TX_CTRL | 0x4000802e | USB host transmission endpoint control register | 0x00 |

USB host physical port control register (R8_UHOST_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| 7 | RB_UH_PD_DIS | RW | Internal pull-down resistor control bit of UD+/UD- pin of USB host port:<br>1: Disable internal pull-down;<br>0: Enable internal pull-down.<br>It can be used in GPIO mode to provide pull-down resistor. | 1 |
| 6 | Reserved | RO | Reserved | 0 |
| 5 | RB_UH_DP_PIN | RO | Current UD + pin status:<br>1: High level; 0: Low level. | X |
| 4 | RB_UH_DM_PIN | RO | Current UD- pin status:<br>1: High level; 0: Low level. | X |
| 3 | Reserved | RO | Reserved | 0 |
| 2 | RB_UH_LOW_SPEED | RW | USB host port low-speed mode enable bit:<br>1: Select 1.5Mbps low-speed mode;<br>0: Select 12Mbps full-speed mode. | 0 |
| 1 | RB_UH_BUS_RESET | RW | USB host mode bus reset:<br>1: Output USB bus reset by force;<br>0: End output. | 0 |
| 0 | RB_UH_PORT_EN | RW | USB host port enable:<br>1: Enable the host port;<br>0: Disable the host port.<br>The bit is automatically cleared to 0 when the USB device is disconnected. | 0 |

USB host endpoint mode control register (R8_UH_EP_MOD)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| 6 | RB_UH_EP_TX_EN | RW | Host transmission endpoint transmit (SETUP/OUT) enable:<br>1: Enable endpoint transmission;<br>0: Disable endpoint transmission. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_UH_EP_TBUF_MOD | RW | Host transmission endpoint transmit data buffer mode control. | 0 |
| 3 | RB_UH_EP_RX_EN | RW | Host reception endpoint reception (IN) enable:<br>1: Enable endpoint reception;<br>0: Disable endpoint reception. | 0 |
| [2:1] | Reserved | RO | Reserved | 00b |
| 0 | RB_UH_EP_RBUF_MOD | RW | USB host reception endpoint reception data buffer mode control. | 0 |

The host transmission endpoint data buffer modes are controlled by a combination of RB_UH_EP_TX_EN and RB_UH_EP_TBUF_MOD, refer to the following table.

Table 17-7 Host transmit buffer modes

| RB_UH_EP_TX_EN | RB_UH_EP_TBUF_MOD | Description: Take R16_UH_TX_DMA as start address |
|---|---|---|
| 0 | X | Endpoint is disabled, and R16_UH_TX_DMA buffer is not used. |
| 1 | 0 | Single 64-byte transmit buffer (SETUP/OUT). |
| 1 | 1 | Double 64-byte transmit buffer, selected by RB_UH_T_TOG:<br>When RB_UH_T_TOG=0, select the first 64-byte buffer;<br>When RB_UH_R_TOG=1, select the last 64-byte buffer. |

The USB host reception endpoint data buffer modes are controlled by a combination of RB_UH_EP_RX_EN and RB_UH_EP_RBUF_MOD, refer to the following table.

Table 17-8 Host receive buffer modes

| RB_UH_EP_RX_EN | RB_UH_EP_RBUF_MOD | Description: Take R16_UH_TX_DMA as start address |
|---|---|---|
| 0 | X | Endpoint is disabled, and the R16_UH_RX_DMA buffer is not used. |
| 1 | 0 | Single 64-byte receive buffer (IN). |
| 1 | 1 | Double 64-byte receive buffer, selected by RB_UH_R_TOG:<br>When RB_UH_R_TOG=0, select the first 64-byte buffer;<br>When RB_UH_R_TOG=1, select the last 64-byte buffer. |

USB host receive buffer start address (R16_UH_RX_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_UH_RX_DMA | RW | Host endpoint data receive buffer start address. The lower 15 bits are valid, and the address must be 4 bytes aligned. | XXXXb |

USB host transmit buffer start address (R16_UH_TX_DMA)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [15:0] | R16_UH_TX_DMA | RW | Host endpoint data transmit buffer start address.<br>The lower 15 bits are valid, and the address must be 4 bytes aligned. | XXXXb |

USB host auxiliary setting register (R8_UH_SETUP)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UH_PRE_PID_EN | RW | Low-speed preamble packet PRE PID enable bit:<br>1: Enable, used to communicate with low-speed USB device through an external HUB.<br>0: Disable the low-speed preamble packet. | 0 |
| 6 | RB_UH_SOF_EN | RW | Automatically generate SOF packet enable bit:<br>1: The host automatically generates SOF packet;<br>0: The host does not automatically generate SOF packet, but can generate manually. | 0 |
| [5:0] | Reserved | RO | Reserved | 000000b |

USB host token setting register (R8_UH_EP_PID)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:4] | MASK_UH_TOKEN | RW | Set the token PID packet identification of this USB transmission transaction. | 0000b |
| [3:0] | MASK_UH_ENDP | RW | Set the endpoint number of the target device being operated this time. | 0000b |

USB host reception endpoint control register (R8_UH_RX_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | RB_UH_R_TOG | RW | Synchronization trigger bit expected by USB host receiver (process IN transaction):<br>1: Expect DATA1;<br>0: Expect DATA0. | 0 |
| [6:5] | Reserved | RO | Reserved | 00b |
| 4 | RB_UH_R_AUTO_TOG | RW | Synchronization trigger bit auto toggle enable control bit:<br>1: After the data is successfully received, the corresponding expected synchronization trigger bit (RB_UH_R_TOG) is automatically toggled;<br>0: It is not toggled automatically, but can be switched manually. | 0 |
| 3 | Reserved | RO | Reserved | 0 |
| 2 | RB_UH_R_RES | RW | Control on response to IN transactions by host receiver:<br>1: No response, used for real-time/synchronous transmission of non-endpoint 0;<br>0: Respond to ACK. | 0 |
| [1:0] | Reserved | RO | Reserved | 00b |

USB host transmission length register (R8_UH_TX_LEN)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| [7:0] | R8_UH_TX_LEN | RW | Set the number of data bytes that USB host transmission endpoint is ready to send. | XXh |

USB host transmission endpoint control register (R8_UH_TX_CTRL)

| Bit | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 7 | Reserved | RO | Reserved | 0 |
| 6 | RB_UH_T_TOG | RW | Synchronization trigger bit prepared by USB host transmitter (process SETUP/OUT transactions):<br>1: Transmit DATA1;<br>0: Transmit DATA0. | 0 |
| 5 | Reserved | RO | Reserved | 0 |
| 4 | RB_UH_T_AUTO_TOG | RW | Synchronization trigger bit auto toggle enable control bit:<br>1: The corresponding synchronization trigger bit (RB_UH_T_TOG) is toggled after the data is sent successfully;<br>0: It is not toggled automatically, but can be switched manually. | 0 |
| [3:1] | Reserved | RO | Reserved | 000b |
| 0 | RB_UH_T_RES | RW | Response control bit of USB host transmitter to SETUP/OUT transaction:<br>1: Expect no response, used for real-time/synchronous transmission of non-endpoint0;<br>0: Expect to respond to ACK. | 0 |

# Chapter 18 Reserved

# Chapter 19 Wireless Communication

## 19.1 Introduction

The chip integrates low-power 2.4-GHz wireless communication modules, including RF transceiver, baseband, link control and antenna matching network. Bluetooth Low Energy (BLE) is supported. More than 100 registers are provided internally to adjust parameters, control process and status. This datasheet does not provide a detailed description of registers. The underlying operations of wireless communication mainly provide application support with subroutine libraries.

Main features:

- Integrated with 2.4GHz RF transceiver, BaseBand and LLE link control.
- Support Bluetooth Low Energy (BLE), compatible with Bluetooth Low Energy specifications of version 4.2/5.0 and above.
- Single-ended RF interface and simplified board design.
- 2Mbps, 1Mbps, 500Kbps and 125Kbps.
- -98dBm RX sensitivity.
- Programmable TX power from -16dBm to +6dBm, and dynamic adjustment is supported.
- When using the PCB onboard antenna, the wide communication distance is about 210 meters at 0dBm TX power, and it is about 500 meters at 6dBm TX power.
- The wide communication distance is about 1000 meters at 6dBm TX power and at 125Kbps.
- Support AES encryption/decryption.
- Support DMA.
- Optimized protocol stack and application layer API, and support networking.

## 19.2 LLE module

LLE module supports automatic sending and receiving mode and manual sending and receiving mode. 5 groups of independent hardware timers can control the time point of sending and receiving data in any one process.

## 19.3 DMA module

The controller has 2 sets of DMA, and each set of DMA has 2 channels. The 2 channels of DMA0 are used to send and receive data respectively, and the 2 channels of DMA2 are used in automatic mode. In automatic transmission mode, the address of sending DMA and the address of receiving DMA can be configured at the same time, so configuration is not needed during the frame interval.

## 19.4 BB module

## 19.5 AES module

Please conduct specific applications based on BLE protocol stack library and refer to the BLE application examples provided.

# Chapter 20 Parameters

## 20.1 Absolute maximum ratings

Stresses at or above the absolute maximum ratings listed in the table below may cause permanent damage to the device.

Table 20-1 Absolute maximum ratings

| Symbol | Parameter description | Min. | Max. | Unit |
|---|---|---|---|---|
| TA | Ambient temperature during operation | -40 | 85 | ℃ |
| TS | Ambient temperature during storage | -40 | 125 | ℃ |
| VDD33 | System supply voltage (VDD33 connects to power, GND connects to ground) | -0.4 | 4.2 | V |
| VIO33 | I/O supply voltage (VIO33 connects to power, GND connects to ground) | -0.4 | 4.2 | V |
| VIO | Voltage on input/output pins | -0.4 | VIO33+0.4 | V |
| VIO5 | Voltage on input/output pins that support 5V tolerant voltage | -0.4 | 5.5 | V |
| VDCI | Voltage on VDCID/VDCIA pin (if external DC-DC is used) | -0.4 | VDD33+0.4 | V |
| VXCK | Voltage on PA10/PA11 after X32MI/X32MO/ enabling LSE | -0.3 | 1.4 | V |

## 20.2 Electrical characteristics

Test conditions: TA=25℃, VDD33=VIO33=3.3V, Fsys=16MHz。

Table 20-2 Electrical characteristics

| Symbol | Parameter description | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| VDD33 | System supply voltage @VDD33 | CH583M | 1.75 | 3.3 | 3.6 | V |
| | | Non-CH583M | 2.3 | 3.3 | 3.6 | V |
| VIO33 | I/O supply voltage @VIO33 | CH583M | 1.75 | 3.3 | 3.6 | V |
| | | Non-CH583M | 2.3 | 3.3 | 3.6 | V |
| $ICC_8$ | Straight-through static supply current Condition: codes run in RAM. Add 5mA if running in Flash. | Fsys=8M | | 2.0 | | mA |
| $ICC_{16}$ | | Fsys=16M | | 2.2 | | mA |
| $ICC_{48}$ | | Fsys=48M | | 6.0 | | mA |
| $IDDC_8$ | Static supply current after DC-DC is enabled Condition: codes run in RAM. Add 4mA if running in Flash. | Fsys=8M | | 1.5 | | mA |
| $IDDC_{16}$ | | Fsys=16M | | 1.6 | | mA |
| $IDDC_{48}$ | | Fsys=48M | | 4.0 | | mA |
| VIL | GPIO low level input voltage | | 0 | | 0.9 | V |
| VIH | GPIO high level input voltage | | 2.0 | | VIO33 | V |
| VIL5 | GPIO that supports 5V tolerant voltage low level input voltage | | 0 | | 0.9 | V |

| | | | | | | |
|---|---|---|---|---|---|---|
| VIH5 | GPIO that supports 5V tolerant voltage high level input voltage | | 2.0 | | 5.0 | V |
| VOL | Output low level voltage (5mA/20mA input current) | | 0 | 0.3 | 0.4 | V |
| VOH | Output high level voltage (5mA/20mA output current) | | VIO33-0.4 | VIO33-0.3 | VIO33 | V |
| IIN | Input current of GPIO floating input | | -3 | 0 | 3 | uA |
| IUP | Input current of GPIO with built-in pull-up resistor | | 25 | 60 | 90 | uA |
| IDN | Input current of GPIO with built-in pull-down resistor | | -90 | -60 | 25 | uA |
| Vref | Voltage on VINTA pin (ADC reference voltage) | | 1.035 | 1.05 | 1.065 | V |
| Vdci | Voltage on VDCID pin after DC-DC is enabled | | 1.18 | 1.3 | 1.38 | V |
| Vlvr | Low-voltage reset threshold | CH583X | 1.3 | 1.5 | 1.7 | V |
| | | Non-CH583X | 1.8 | 2.05 | 2.3 | V |

## 20.3 Power consumption in low-power modes

Test conditions: TA=25°C, VDD33=VIO33=3.3V, Fsys=16MHz.

Table 20-3 Low power parameters (for reference only, related to temperature)

| Low-power mode | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|
| Idle mode, enable the clock combination of each module | 1.2 | 1.6 | 1.8 | mA |
| Halt mode, disable all clocks | | 320 | | uA |
| Sleep mode, multiple combinations, refer to Table 5-3 | | 0.7~2.8 | | uA |
| Sleep mode, PMU+core+RAM2K, GPIO wakes up, no RTC | | 0.7 | | uA |
| Shutdown mode, several combinations, refer to Table 5-3 | | 0.2~2.3 | | uA |
| Shutdown mode, only PMU, reset after GPIO wakes up, no RTC | | 0.2 | | uA |

Table 20-4 Current on modules (for reference only, related to temperature)

| Symbol | Parameter description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $I_{DD(RAM2K)}$ | RAM2K:2KB SRAM | | 0.3 | | uA |
| $I_{DD(RAM30K)}$ | RAM30K:30KB SRAM | | 1.4 | | uA |
| $I_{DD(LSI)}$ | Internal LSI oscillator | | 0.3 | | uA |
| $I_{DD(LSE)}$ | External LSE oscillator | | 0.4 | | uA |
| $I_{DD(HSE)}$ | External HSE oscillator | 100 | 200 | 300 | uA |
| $I_{DD(BM)}$ | Low-power battery low-voltage monitor module (BM) | | 0.9 | | uA |
| $I_{DD(BD)}$ | High-precision battery low-voltage detector module (BD) | | 100 | | uA |
| $I_{DD(PLL)}$ | Internal PLL oscillator | | 150 | | uA |
| $I_{DD(ADC)}$ | ADC module | | 0.4 | | mA |
| $I_{DD(TKEY)}$ | Touch-Key module | | 0.1 | | mA |

| | | | | | | |
|---|---|---|---|---|---|---|
| $I_{DD(TS)}$ | Temperature sensor module (TS) | | | 0.1 | | mA |
| $I_{DD(USB)}$ | USB module | Not in transmit status | 1.2 | 1.6 | 2.0 | mA |
| | | Transmit status | | 3 | | mA |
| $I_{DD(BLE)}$ | BLE | Receiving | Direct power supply | 13.5 | | mA |
| | | | Enable DC-DC | 6.5 | | mA |
| | | -16dBm transmission power | Direct power supply | 5 | | mA |
| | | | Enable DC-DC | 2.5 | | mA |
| | | 0dBm transmission power | Direct power supply | 10 | | mA |
| | | | Enable DC-DC | 4.5 | | mA |
| | | +6dBm transmission power | Direct power supply | 15.5 | | mA |
| | | | Enable DC-DC | 7 | | mA |

## 20.4 Clock source

Table 20-5 High-speed oscillator (HSE)

| Symbol | Parameter description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $F_{HSE}$ | External HSE oscillator frequency | | 32 | | MHz |
| $T_{SUHSE}$ | External HSE oscillator startup to available time | 80 | 200 | 500 | uS |
| $T_{STHSE}$ | External HSE oscillator startup to stabilization time | 200 | 500 | 3000 | uS |

Table 20-6 Low-speed oscillator (LSI and LSE)

| Symbol | Parameter description | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $F_{LSIR}$ | Internal LSI oscillator frequency (before calibration) | | 20K | 32K | 48K | Hz |
| $F_{LSI}$ | Internal LSI frequency (after application software runtime calibration) | | 32726 | 32768 | 32810 | Hz |
| $A_{LSI}$ | LSI oscillator accuracy (after software calibration) | TA=-40℃~85℃ | | 0.1 | 0.5 | % |
| | | TA=0℃~60℃ | | 0.04 | 0.2 | % |
| $T_{SULSI}$ | Internal LSI oscillator startup to stabilization time | | | 40 | 100 | uS |
| $T_{SULSE}$ | External LSE oscillator startup to available time | | 100 | 300 | 1500 | mS |
| $T_{STLSE}$ | External LSE oscillator startup to stabilization time | | 500 | 1500 | 5000 | mS |

Table 20-7 PLL characteristics

| Symbol | Parameter description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $F_{PLL}$ | Output clock after PLL (CK32M * 15) | | 480 | | MHz |
| $T_{PLLLK}$ | PLL lock time | | 15 | 30 | uS |

## 20.5 Timing parameters

Test conditions: TA=25℃, VDD33=VIO33=3.3V, Fsys=6.4MHz.

Table 20-8 Timing parameters

| Symbol | Parameter description | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $T_{rpor}$ | Reset delay after RPOR | | 11 | 15 | 20 | mS |
| $T_{rst}$ | RST# valid signal width | | | 100 | | nS |
| $T_{mr}$ | Reset delay after MR | | 2 | 8 | 18 | uS |
| $T_{sr}$ | Reset delay after SR | | 2 | 8 | 18 | uS |
| $T_{wtr}$ | Reset delay after WTR | | 10 | 12 | 18 | uS |
| $T_{WAK}$ | Wakeup time to exit from low-power status | Idle Mode | 0.6 | 1 | 3 | uS |
| | | Halt Mode | $T_{SUHSE}$+1 | $T_{SUHSE}$+80 | $T_{SUHSE}$+150 | uS |
| | | Sleep Mode | $T_{SUHSE}$+1 | $T_{SUHSE}$+300 | $T_{SUHSE}$+400 | uS |
| | | Shutdown Mode | $T_{SUHSE}$+0.4 | $T_{SUHSE}$+1 | $T_{SUHSE}$+5 | mS |

*Note: The delay parameters in the table are all based on multiples of Tsys, and the delay will be increased when the clock frequency is reduced.*

*The delay parameters in the table are based on using an external HSE clock source. If an external HSE clock source is used during sleep, the delay parameter $T_{WAK}$ in Halt mode/Sleep mode/Shutdown mode in the table will be increased by about 0.2~1mS (activated to available $T_{SUHSE}$).*

## 20.6 Other parameters

Test conditions: TA=25℃, VDD33=VIO33=3.3V, Fsys=16MHz.

Table 20-9 Other parameters

| Symbol | Parameter description | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $R_{TS}$ | Measurement range of temperature sensor | | -40 | | 85 | ℃ |
| $A_{TSC}$ | Measurement error of temperature sensor after calibrated by software | | | ±10 | | ℃ |
| $T_{FRER}$ | Single sector erase operation time of Flash-ROM | | 6 | 16 | 30 | mS |
| $T_{FRPG}$ | Single word program operation time of Flash-ROM | | 1 | 2 | 4 | mS |
| $N_{EPCE}$ | Erase/program cycle endurance | 5~45℃ | 100K | 1000K (Random test) | | times |
| | | -40~85℃ | 50K | 200K (Random test) | | |
| $T_{DR}$ | Data hold capability of Flash-ROM | | 20 | | | years |
| $V_{ESD}$ | ESD tolerant voltage on I/O pins | Antenna (ANT) | 2K | 4K (Random test) | | V |
| | | I/O pins: PA and PB | 4K | 6K (Random test) | | V |

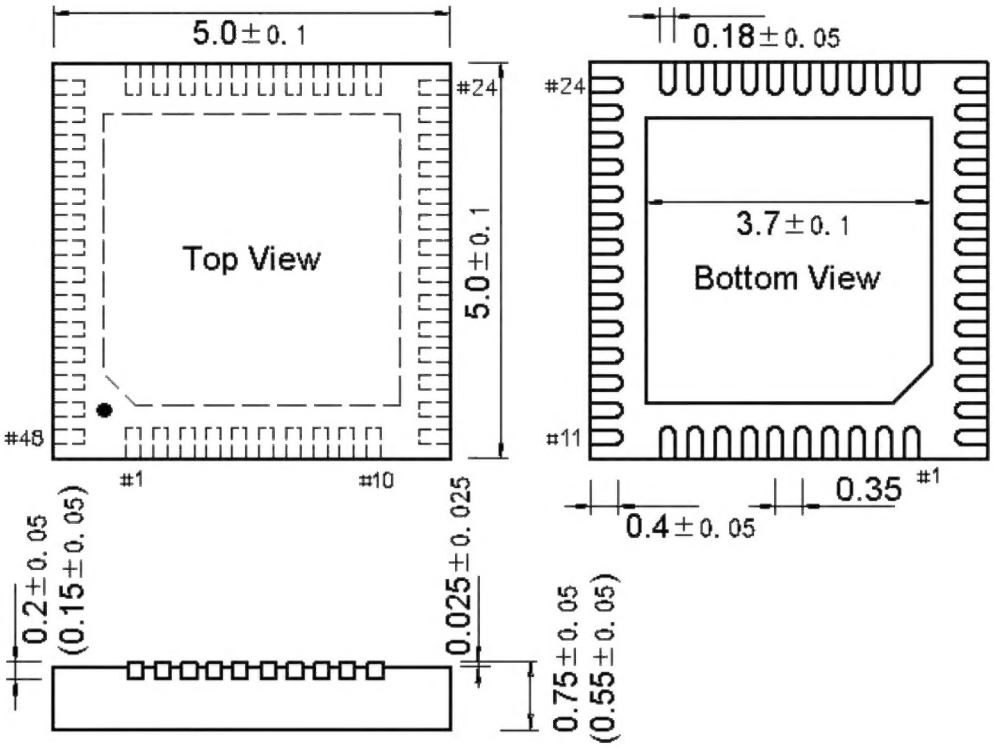
# Chapter 21 Package

Packages

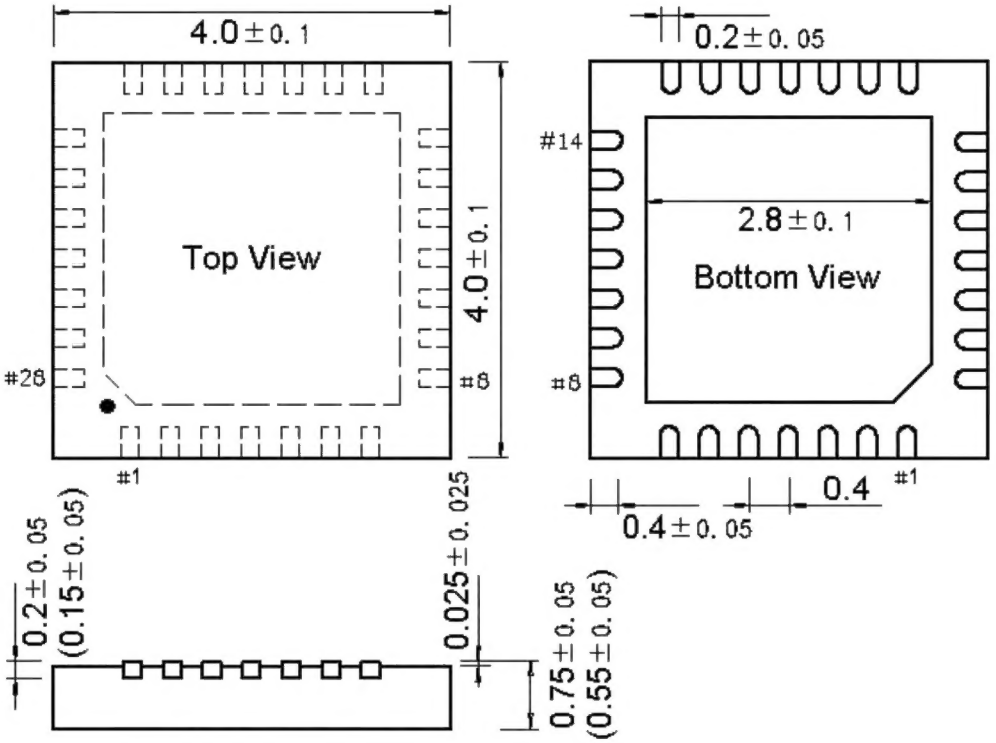
| Package Form | Body Size | Pin Pitch | | Package Description | Order Model |
|---|---|---|---|---|---|
| QFN48 | 5*5mm | 0.35mm | 13.8mil | Quad Flat No-Lead Package | CH583M |
| QFN48 | 5*5mm | 0.35mm | 13.8mil | Quad Flat No-Lead Package | CH582M |
| QFN28 | 4*4mm | 0.4mm | 15.7mil | Quad Flat No-Lead Package | CH582F |

*Note: All dimensions are in millimeters. The pin center spacing values are nominal values, and the error of other dimensions is not more than ±0.2mm.*

QFN48_5×5



QFN28_4×4


4.0±0.1
Top View
4.0±0.1
#28
#8
#1
0.2±0.05
(0.15±0.05)
0.025±0.025
0.75±0.05
(0.55±0.05)
0.2±0.05
#14
2.8±0.1
Bottom View
#8
#1
0.4
0.4±0.05

# Chapter 22 Revision History

| Version | Date | Description |
|---|---|---|
| V0.9 | 2020.05.11 | First release |
| V1.0 | 2020.12.22 | Add section 3.4.2 and 3.4.3 and modify 3.4.1 interrupt and 3.4.4 system counter description, fine tune Bluetooth receive sensitivity, prompt touch detection support driver shield, fine tune sleep current. |
| V1.1 | 2021.06.22 | Characteristic change of R8_GLOB_RESET_KEEP in Section 4.1 added, it is reset when shutdown/GRWSR occurs. |
| V1.2 | 2021.08.10 | Typos in Section 20.2 corrected. Package figure of QFN48 updated (by default, the thickness is 0.75mm) |
| V1.3 | 2021.09.23 | RB_WAKE_DLY_MOD updated. Package figures updated, with size marked on the figures. |
| V1.4 | 2022.01.21 | Typos (I2C_CTRL) corrected. Transmission power limited up to +6dBm. PB12 and PB13 do not support 5V. For auxiliary power adjustment control register (R16_AUX_POWER_ADJ), it should be 16-bit if write operation is needed. |
| V1.5 | 2022.05.24 | CH583M memory capacity updated to be the same as that of CH582M. |
| V1.6 | 2022.06.27 | Typos corrected: RB_ADC_BUF_EN should be set to 0 for temperature sensor sampling. |
| V1.7 | 2022.10.18 | Correct RB_XT32K_C_LOAD description, external HSE frequency delete non-32MHz |
| V1.8 | 2024.01.08 | Modify the capacitance value recommendation in the VINTA pin description and delete the FLASH erase and write procedure in section 4.4. |
| V1.9 | 2025.04.10 | Fix multiple clerical errors in Tadc, SPI crossover, etc., add description of I2C_CKCFGR, delete CH581F. |
| | | |